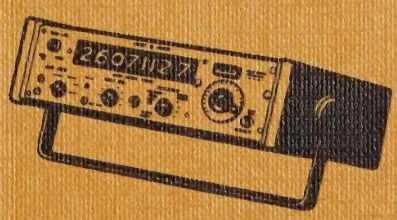
An Introduction to

DIGITAL ELECTRONICS

Theory, Instruments and Computers

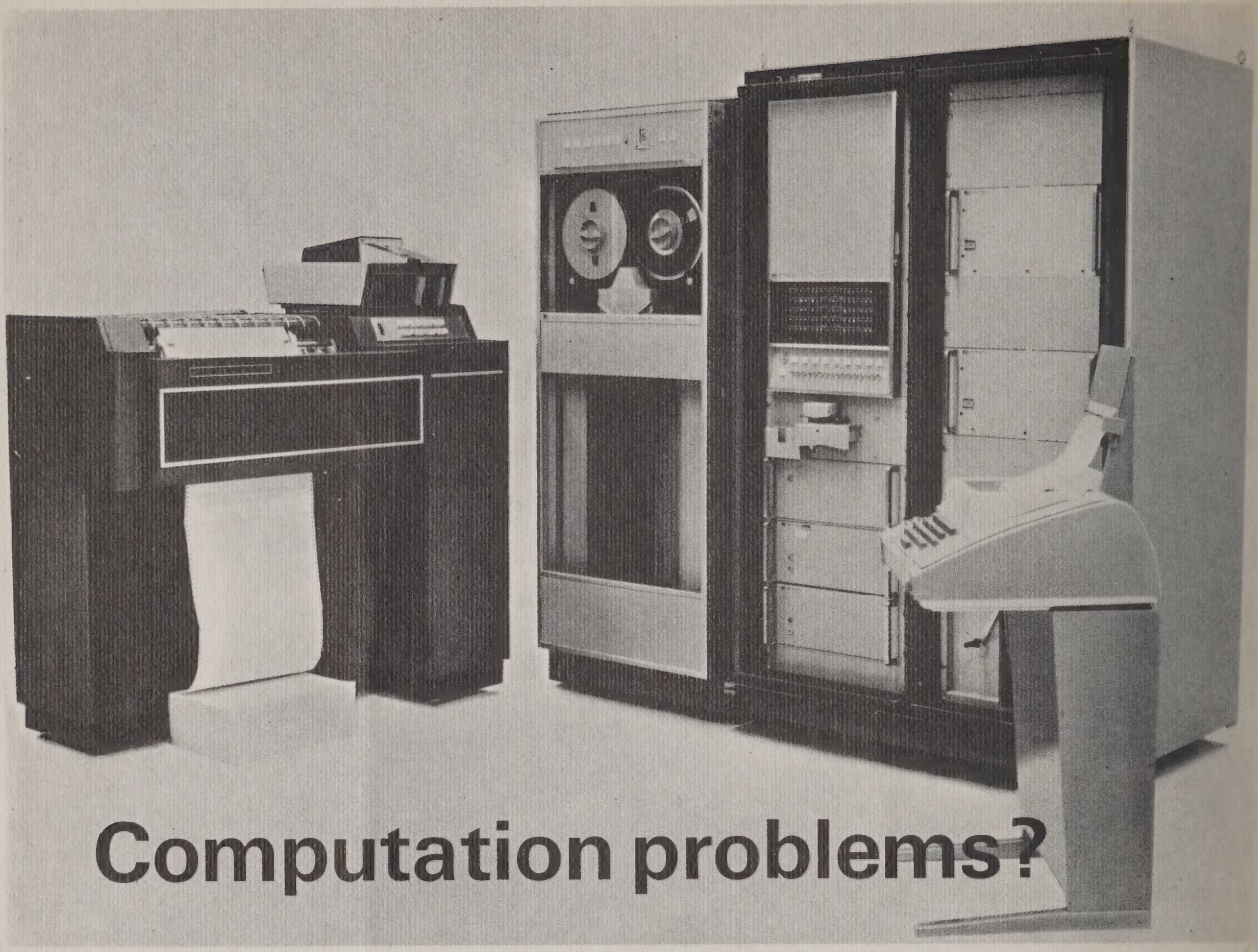
by Jamieson Rowe

An "Electronics Australia" Publication



SECOND EDITION, 1970

PRICE $2.00



Computation problems?

...then there is an HP System for you.

Systems are available for data acquisition, batch processing, time sharing, real time, education.

Illustrated is the HP 2005A Real-Time Executive (RTE), a true, real time multi-programming system built on the proven HP 2116B Computer and its standard peripherals and options.

The key to the RTE system is the Executive software which schedules the running of programmes on a priority basis under both time and event control. By supervising all interrupts, 1/0 operations, bulk memory transfers and housekeeping, the Executive permits several programmes to run in real time concurrently with general purpose background programmes.

To the user this means maximum utilisation of hardware and a practical method of obtaining multiple data processing capabilities where separate computers are not economically feasible.

Real time software includes an assembler, FORTRAN compiler, symbolic editor, debug programme, relocating loader and library. The user can write programmes in HP Real-Time Assembly Language or HP Real-Time FORTRAN.

For further information on the HP Systems contact the Hewlett-Packard Sales Office in your area. Melbourne: 22-26 Weir Street, Glen Iris. 3146. Sydney: 61 Alexander Street, Crows Nest. 2065. Canberra: 10 Woolley Street, Dickson. 2602. Perth: 196 Adelaide Terrace. 6000. Auckland (N.Z.): P.O. Box 51092 Pakuranga. Wellington (N.Z.): P.O. Box 9443, Courtenay Place.

HEWLETT hp PACKARD

HP43

Printed and published by Sungravure Pty. Limited, of Jones Street, Broadway, at Morley Avenue, Rosebery.

An Introduction to

DIGITAL ELECTRONICS

Theory, Instruments and Computers

by Jamieson Rowe
B.A.(Sydney), B.Sc.(Technology, N.S.W.)
A.M.I.R.E.E. (Aust.)
Technical Editor, "Electronics Australia"

An "Electronics Australia" Publication
Printed and published by Sungravure Pty. Limited,
of Jones Street, Broadway, at Morley Avenue,
Rosebery, New South Wales.

Foreword to Second Edition

No doubt it is almost always a relatively pleasant task to write introductory remarks for later editions and print runs of a publication, because the necessity for reprinting generally implies that a work has encountered at least a modest degree of success. However in this instance the task is a particularly pleasant one, as the response met by the first edition has been very gratifying.

That the initial printing of such a relatively specialised technical book has been exhausted in a bare three-year period is perhaps in itself a fact worthy of some gratification. However of greater significance to the author have been the many compliments received from readers, and the communications from technical colleges and other educational bodies informing him that book has been adopted as either a course text or commended reference. These seem to suggest that the original aims of the book have been at least partially achieved.

In preparing the book for reprinting we have taken the opportunity to correct a small number of minor typographic and similar errors which have been found since the original printing. An explanatory diagram has also been added to chapter 13, on page 103, as some readers found it difficult to follow the text description of the demonstrator connections required for DFM configurations.

A possible criticism of the book in its present form is that the "discrete" circuitry of the described demonstrator unit must inevitably appear obsolete in comparison with the highly "integrated" circuitry found in modern digital equipment. Yet while it is true that a demonstrator unit employing high performance microcircuits would very likely be preferable for design work and for the demonstration of digital system design, the very nature of microcircuit elements would surely give the reader little more than a "black box" conception of the actual circuitry involved.

As the book is after all a basic text, we have retained the material describing the original demonstrator in the hope that this will, if nothing else, acquaint the reader with some simple circuit configurations capable of performing the same function as the microcircuit elements with which he should later become familiar.

—J.R. (1970-06-09)

Preface

The book which follows is based largely upon a series of articles which the author wrote in "Electronics Australia" magazine, commencing in mid-1966.

The motivation to write the articles arose from the experience of the author when he sought to find a basic introduction to digital circuit techniques. Although he found no shortage of journal articles, manufacturer's data sheets and application notes on specific aspects of the subject, a fairly lengthy search through both technical bookstores and publishers' catalogues failed to discover more than a very few books, none of which seemed to provide a satisfactory treatment of the subject as a whole. The books which were found certainly provided useful information but, in virtually every case, they covered only one or two aspects of the overall picture.

There were texts on symbolic logic and Boolean algebra, very often excellent as such, but in many cases rather unhelpful when one was concerned with gaining an insight into the exact nature of the logic-electronics relationship. In contrast with these were the texts on switching and pulse electronics, which seemed all too frequently to skim hurriedly over Boolean fundamentals, thereafter discussing circuit functions with only a vague and somewhat perfunctory reference to logic. And thirdly there were texts on computers and data processing which, in some cases, gave very useful information in the introductory chapters, but which almost invariably became rapidly involved in specialised aspects of data processing.

What seemed to the author to be needed was a book or a series of articles which would bring together the various aspects of digital electronics, ordering and correlating them into some sort of unity. Only a work written with this aim in view seemed likely to provide the engineer, technician or student with a really clear introduction to the techniques in which logical and electronic concepts are combined so fruitfully.

Reader response to the early articles in the series was encouraging, and grew more so as the later articles followed. The response grew sufficiently favourable to suggest that a book might be justified. Tentative discussions with engineers, technicians and college lecturers lent support to this idea, and also proved most effective in bringing to light various inadequacies of the articles which might well be remedied in the adaptation of the material to book form.

Even more so than with the articles, the aim in producing the book has been to provide an introduction which, while basically qualitative, will be of value not only to the technician and advanced hobbyist but also to the engineer and undergraduate college student. To this end, particular effort has been directed toward establishing and developing a framework of logical concepts which might act as a foundation for further detailed study.

In concluding this preface, the author would like to thank the many people to whom he is indebted for the advice and assistance which will have contributed largely to whatever value the book may be judged to possess. Deserving particular mention in this regard are the Editor and staff of "Electronics Australia."

—Jamieson Rowe.

Contents

Chapter 1

LOGIC and CIRCUITS

An historical review—why logical concepts are used—circuits and statements, truth and falsity—the elementary operations AND, OR and NOT—logic symbols and truth tables—the "building block" operations NOR and NAND.

An integral part of science and technology consists of quantitative description of reality — measurement — and development in these disciplines is largely a matter of refinement of measurement techniques in order to improve validity, reliability and accuracy. For the scientist to describe adequately the nature of reality, he must have an accurate picture of the phenomena concerned. Similarly, for the technologist to bring about desired changes in materials or events, he in turn must have a means of determining accurately the results of his efforts.

It is no accident that the present rapid and constantly accelerating growth in science and technology correlates highly with the growth of electronics, one of science's most recent "offspring" disciplines, for a highly important application of electronic techniques has been in the refinement of measurement in almost every field of scientific and technological activity.

Probably the most significant development in electronics in recent years has been that of digital technology—a set of techniques whereby equipment can perform tasks of measurement, monitoring, controlling and signal processing with a reliability, convenience and degree of precision far beyond the limits realisable by previous techniques. In a few short years digital technology has not only produced tremendous advances in existing fields such as measurements, communications and automatic control, but it has virtually created a whole new field which seems destined to make profound changes in human culture as a whole: Computers and data processing.

While computers and data processing are but one of the topics to be treated in this book, some knowledge of the development of this field is relevant at the outset because, to a large extent, its development coincides with that of digital technology as a whole. In any case it was largely the need **for** computers and data processing facilities which prompted the development of digital technology; while it is probably also true to say that it is the prodigious benefits being reaped increasingly **from** this application which provide a major part of the continuing motivation for further development.

To gain some insight into the growth of digital techniques as a whole it will therefore be worthwhile to review briefly the development of computers and data processing.

Almost since the dawn of their art, it seems, mathematicians have dreamed of having machines which would be able to free them from the drudgery of tedious repetitive calculations. Although this dream is only being realised fully in the twentieth century, modern computing machines are by no means without precedent.

Probably the first to invent a computing machine was the sixteenth-century Scottish mathematician John Napier (1550-1617), better known for his invention of **logarithms;** his so-called "Napier's Bones" was a crude multiplication machine. Napier also was very likely the first "modern" mathematician to conceive the abstract importance of the binary or **base-2** numerical scale—which we will see later to play an important part in the operation of most modern digital equipment.

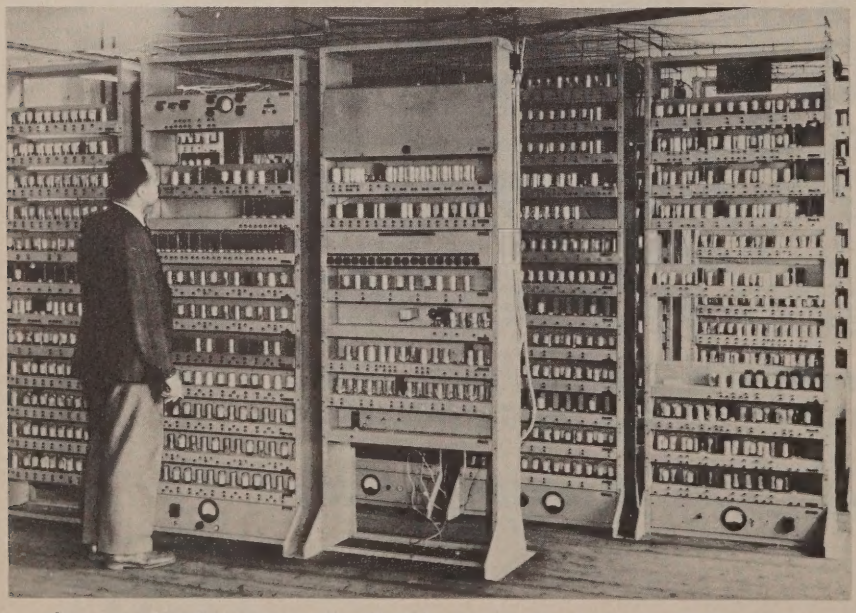
At the age of 18 the brilliant French mathematician Blaise Pascal (1623-1662) invented what was probably the first adding machine or "desk calculator" to assist him in the job of keeping his father's business accounts.

In turn, Pascal's machine was improved upon by the great German philosopher-mathematician Gottfried Wilhelm von Leibnitz (1646-1716), who invented a machine which could perform counting, addition, subtraction, multiplication and division. Leibnitz realised quite well the advantages of the binary or base-2 system over the familiar decimal system where "machine" arithmetic was concerned.

Although these early calculating machines were important steps in the development of modern digital computers and instruments, in themselves they did little more than simply point the way in which further developments might take place. For the really significant developments could not take place until important discoveries had been made—not only in engineering, but in a subject apparently unconnected with the problem. That subject was **logic.**

Until about the middle of last century logic had remained somewhat aloof from mathematics as a traditional part of the "higher" discipline of philosophy. First presented in an organised fashion by the great Greek philosopher Aristotle (384-322 B.C.), it had remained essentially unchanged through ancient and medieval times as an "ivory tower" pursuit of philosophers and a tool for the justification of theological doctrines.

However the estrangement of logic from mathematics began to dissolve early last century with the publication in 1847 of the works of two important logicians. In his book **An Investigation Of The Laws Of Thought,** George Boole (1815-1854) laid the foundation for a complete symbolism of logic; while Augustus de Morgan (1806-1872) provided extensive evidence that logical validity was not confined solely to the traditional or "Aristotelian" modes of inference.



One of the first electronic digital computers, built in 1949 at the Mathematical Laboratory of Cambridge University. It was known as "EDSAC" (Electronic Delay Storage Electronic Calculator).

The work of Boole and de Morgan was developed further by such logicians and mathematicians as William S. Jevons, Charles Peirce, John Venn and C. L. Dodgson ("Lewis Carroll"). Finally, using and extending the arguments put forward by the German philosopher Gottlob Frege, the Cambridge logicians Bertrand Russell (b. 1872) and Alfred North Whitehead (1861-1947) published in 1910-1913 their monumental work **Principia Mathematica.** In this they demonstrated conclusively that logic is, in fact, the basis upon which mathematics rests.

Along with the developments in logic and mathematics the nineteenth century brought advances in technology. The Englishman Charles Babbage (1792-1871) invented the first really modern calculating machine in 1835. Called by him an "analytical difference engine," it worked in decimals and could make continual logical decisions throughout a predetermined sequence of operations. It could add 29-digit numbers and if necessary carry over all 29 digits simultaneously!

Babbage's machine was entirely mechanical since in his day electrical engineering was still in its infancy. It was not until the 1930s that the first electrical logic machine was reported to have been built by Benjamin Burack of Roosevelt College in Chicago.

The final step which was to make possible the development of modern digital technology was made in 1937 by Claude E. Shannon, in his thesis submitted at the Massachusetts Institute of Technology for the award of M.Sc. In the thesis Shannon showed that Boole's symbolic logic could be used, not only as a means of designing extremely efficient circuits capable of carrying out mathematical operations, but also in the organisation or "programming" of the operations to be carried out by the circuits.

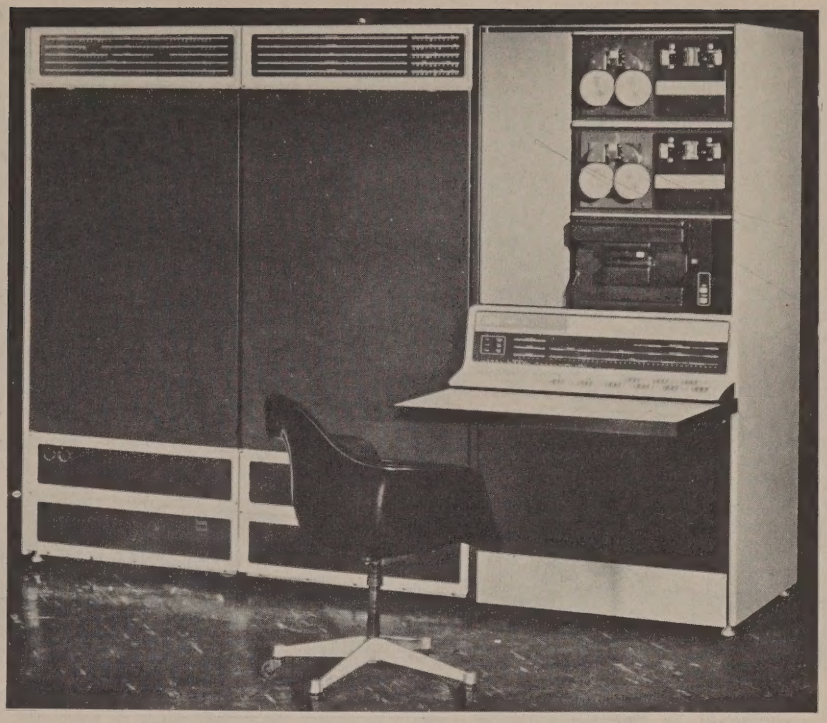
It had occurred to many before Shannon that electrical circuits might be used to carry out mathematical operations. However, it had remained for Shannon to show that the concepts of symbolic logic provided the most efficient way of designing such circuits. From this final bringing-together of the disciplines of mathematics, logic and electrical engineering, digital technology was born.

In the late 1940s, the "Mark I" electro - mechanical digital computer was produced at Harvard University. Soon after, laboratories in both England and America produced fully electronic computers. In the 1950s commercial production began, and since then there has been a continuous and accelerating stream of machines that are more efficient, faster, more versatile or more specialised than their predecessors.

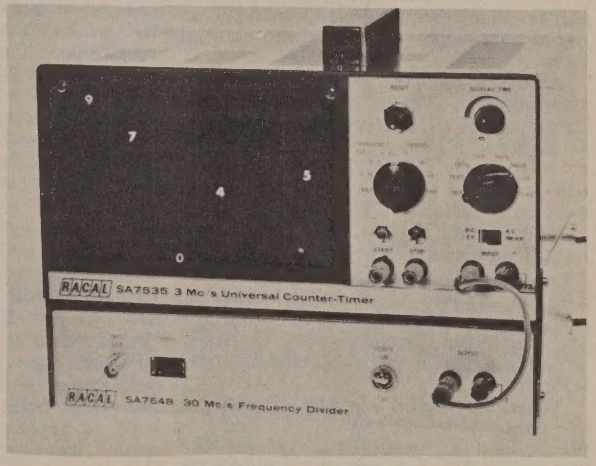
Right from the start the techniques developed initially for computing applications were generalised and extended into other fields, to form the roots of the new discipline. Almost immediately after the release of the first commercial computers came the release of the first digital measuring instruments. Since then, these too have formed an ever-increasing stream, accompanied by an equally rapid extension of digital techniques into almost every sphere and application of electronics.

It is hoped that the foregoing brief historical review will help the reader in understanding the factors which contributed to the recent appearance of digital computers and instruments. It should also show why the concepts of symbolic logic or "Boolean algebra" have become a part of the knowledge required for a full understanding of the operation of these machines.

In view of the important part played by symbolic logic in the operation of digital circuitry we may, with profit, commence the study of these circuits with a description of the elementary logical operations.



A modern high-speed general purpose computer designed for a wide variety of applications in scientific research and industry, the "PDP-10" manufactured by Digital Equipment Corporation.

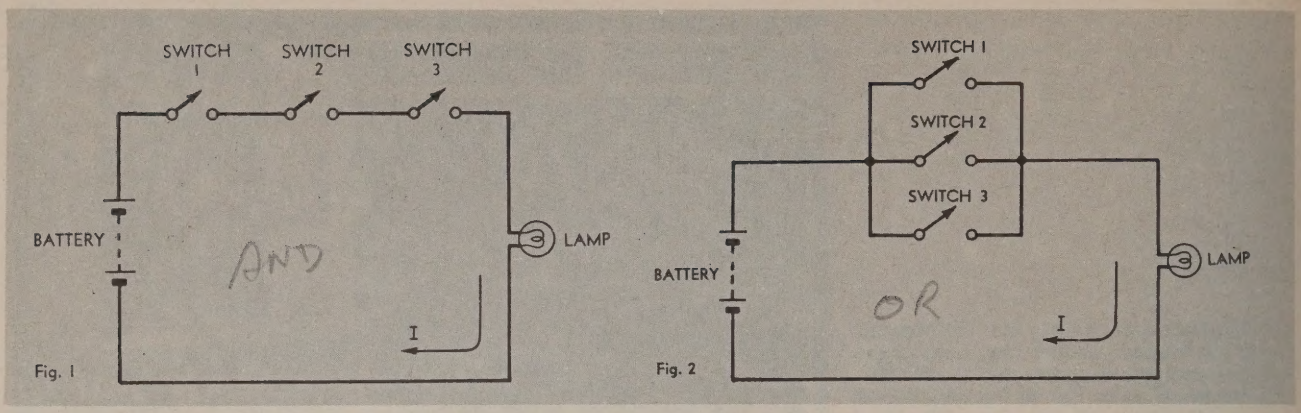


A modern digital frequency meter and timer manufactured by Racal Electronics Pty. Ltd. It measures up to 30MHz with an accuracy of 2 parts per million.

There are three elementary logical operations or "functions" from which virtually all logical circuitry is derived. These are usually known as the AND, OR and NOT operations. Other names for the AND operation are **logical product** and **conjunction,** and for the OR operation **logical sum** and **disjunction.** Alternative names for the NOT operation are **logical negation, inversion** and **complementation.**

Any circuit or device which effectively delivers an output only when two or more specified input quantities occur simultaneously may be interpreted as performing the logical AND operation, in that it is exhibiting a response to a **conjunction** of input quantities.

For an example of a simple AND element consider the simple switch and lamp bulb circuit of figure 1. Here it is fairly obvious that a current "I" will flow through the lamp only if **all three** of the switches are closed together. If any of the switches is open, the lamp cannot light.



Simple circuits used to introduce basic circuit logic concepts.

If one were asked to describe the **electrical** operation of this circuit in terms of functional "inputs" and "outputs", one would probably reply that it has three inputs which correspond to the closure of the switches, together with an output corresponding to the lighting of the lamp.

However, from a **logical** viewpoint the input and output quantities of a circuit element are not the electrical states-of-affairs themselves, but **statements** or propositions which they are used to represent. The operation of the element is thus interpreted in terms of **logical** relationships between these statements rather than electrical relationships between the states-of-affairs representing them.

From a strictly logical viewpoint, statements can have only two possible "truth values" if they are to be regarded as meaningful: they must be either **true** or **false.** They can neither be both true-and-false nor anything other than either true or false.

Having only two possible truth values, logical statements are well suited to be represented by electrical states-of-affairs. This is because it is much easier to design reliable **two-state** electrical and electronic circuitry than to design circuitry which must give reliable multi-state operation.

The three **logical** inputs to the circuit of figure 1 could be symbolised as A, B, and C, where the letters "A," "B" and "C" are symbols standing for logical statements such as "The doors are closed," "The lift is not overloaded," and "A destination is registered."

If this is done the **closure** of switch 1 may be used to represent the **truth** of statement A, so that conversely if switch 1 is **open** it would represent statement A as being false. Similarly the closure of switches 2 and 3 may be taken to represent the truth of statements B and C, with the falsity of either statement represented by the open condition of its corresponding switch.

The logical **output** of the circuit may be interpreted as a fourth statement symbolised by Z. Here the flow of current I and the lighting of the lamp may be taken to represent the **truth** of Z (i.e., "Start lift motor"), and the absence of these phenomena to represent its falsity.

Fairly obviously, the action of the circuit is such that statement Z will only be represented as true (lamp alight) when all three of the input statements are represented as true together (all switches closed). We can therefore represent the **logical** operation of the circuit by the symbolic relationship

$$A.B.C = Z \qquad \ldots(1)$$

Here the dots between the symbols are used to symbolise the logical AND (which can alternatively be represented by a product sign "×"), and the equals sign symbolises logical equivalence. In words the relationship therefore expresses the proposition:

> **"If statement A is true AND statement B is true AND statement C is true, then statement Z is also true."**

Although in the above example the **closure** of the input switches was used to represent the **truth** of the corresponding input statements, and the flow of current I to represent the truth of the output statement, it is by no means necessary in logic circuitry to have this fixed relationship. The correspondence between the truth and falsity of logical statements and the electrical states-of-affairs used to represent them is, in fact, quite an arbitrary one, and may be assigned and even varied as required.

Thus the truth of a statement may be represented equally validly by **either** the open or closed state of a switch, the presence **or** absence of a voltage or current, the positive **or** the negative polarity of a voltage, and so on. But note that whichever electrical state is used to represent the truth of a statement, its opposite will automatically become representative of the statement's falsity.

The particular logical-electrical equivalence being used at any one time or place in a logical network is termed the **logic convention.** As will be shown later, manipulation of the logic convention plays an important part in the process of simplifying the circuitry required to perform a complex logic operation.

Consider in the case of figure 1 what would happen if the closure of switch 2 were taken to represent the **falsity** of a statement B, rather than its truth. Then for statement Z to be true, statements A and C would have to be true, while at the same time statement B would have to be false. This is the same as saying that statements A, **non**-B and C must be true; here "non-B" signifies that statement which is true when B itself is false—i.e., the contradiction or **complement** of B.

We can represent the new situation by the symbolic relation

$$A.\bar{B}.C = Z \qquad \ldots(2)$$

where the "bar" over the symbol "B" is taken to signify logical inversion or complementation. In words the relation expresses the proposition:

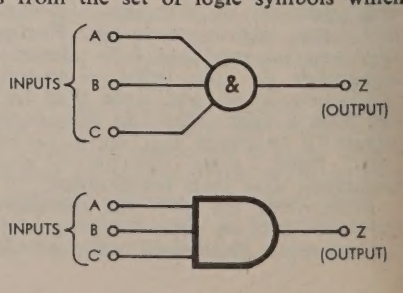
> **"If statement A is true AND statement non-B is true AND statement C is true, then statement Z is also true."**

It may seem a little strange that it is necessary in the above example to convert the falsity of statement B into the truth of its complement non-B, before the operation of the circuit can be defined again as an AND operation. However the simple answer to this is that the elementary logical operation AND is **by definition** one in which the output statement is **true** only when all the input statements are also **true.** Logically it is therefore irrelevant whether or not one or more of the statements actually concerned in the AND situation happens to be the complement of those to which we have given the labels "A," "B," "C," and so on; all that is relevant is that the output statement is true when there is the stipulated number of true input statements.

By definition it is therefore **not possible** to have an elementary AND operation where the output statement is true when there is a certain combination of true and false input statements. In fact such a situation is a complex one which combines the AND operation with the NOT operation to be discussed shortly. Similarly one cannot have an AND operation in which the output is **false** when the inputs are all true; this is also a complex logical situation, as we will see.

Just as it is possible to draw schematic circuit diagrams which symbolise the **electronic** function of the various elements in a circuit — figure 1 is an example of such a diagram — it is also possible to draw schematic **logic** diagrams which symbolise the function of elements in a logic network.

The symbols used in such diagrams to represent an AND circuit or device are shown below. The upper symbol is from the set of logic symbols which



have become standardised in England, Europe and Australia (B.S. 530:1948, Suppl. No. 5, 1962), while below it is the symbol from the set approved for U.S. military use (MIL STD 806B). The former symbols will be used throughout this book, as the author believes that they are more readily recognised and distinguished from one another.

The logical operation performed by a device or circuit may be **defined** rather conveniently, and very concisely, in terms of all the possible combinations of truth and falsity of the input statements and the corresponding truth values of the output statement. The usual way of defining a logic element or network in this fashion is by means of a so-called **"truth table."**

The truth table definition for a three-input AND element or "gate" is shown below. There are four vertical columns representing the input and output statements, and eight horizontal rows representing all the possible combinations of the input statement truth values together with the corresponding output truth values. In each case **truth** is shown by a "1" and **falsity** by a "0."

INPUTS			OUTPUT
A	B	C	Z = A.B.C
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

And

As may be seen the truth table shows clearly that the output of an AND gate is true only when all (here, all three) of the inputs are also true. For all other combinations the output is false.

It should perhaps be emphasised at this juncture that the AND operation is not restricted to three inputs, but may in fact involve any number greater than one. Three inputs have been used in the above example purely for convenience of illustration.

Consider now a second circuit, that shown in figure 2. Here we again have three switches connecting a battery to a lamp bulb, but the switches are now in parallel. Instead of the previous case where all three were required to be closed before the lamp would light, we now have a situation where any switch alone can allow the current I to flow.

The new circuit can be used to perform the logical OR operation, because current flows if switch 1 is closed OR switch 2 is closed OR switch 3 is closed.

Any circuit or device may be said to perform the logical OR operation if it delivers an output whenever one or more input quantities are present. Or more strictly and in terms of truth values, the OR operation is defined as one wherein the output statement is true provided that **at least one** of the input statements is true. As **any** one—or more—of the input statements may be true, the situation is one of logical **disjunction.**

Because the definition of an OR element stipulates only that **at least** one of the inputs must be true for the output to be true, the output of such an element can and will be true when **more than one** of the inputs are true. It follows that one possible input combination which will result in a true output is the "all true" or AND situation.

MYRIAD II, the first commercially available digital computer constructed from integrated microcircuits. Although small, it is both fast and versatile. (Courtesy Marconi Company Ltd.)

Hence from a logical viewpoint the OR operation **includes** the AND operation.

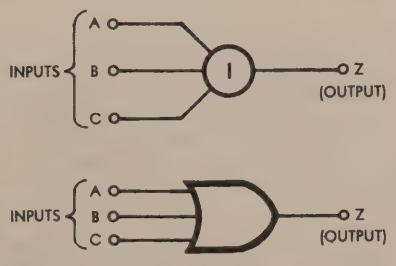
It is logically quite feasible (and electrically quite practical) to have a situation where the output is true when **one and only one** of the inputs is true, and for this reason the OR operation is sometimes called the "inclusive-OR" operation to distinguish it from such a situation. However, this is not really necessary as the one-and-only-one operation is **always** distinguished by the name "exclusive-OR."

If the symbols "A," "B" and "C" are used as before to represent the three input statements whose truth is represented by the closure of the switches, and "Z" is used to represent the output statement whose truth is represented by the lamp lighting, the logical operation of the circuit of figure 2 may be shown by the relationship

$$A + B + C = Z \qquad \ldots(3)$$

Here the algebraic "plus" signs are used to symbolise logical OR, and the equals sign again symbolises logical equivalence.

The logical symbols for the OR operation are shown below. The upper symbol is that to be used throughout this book; the lower is the U.S. Military Standard symbol.



As with the AND operation, the OR operation may be defined in terms of all the combinations of truth and falsity of the input statements, and the corresponding values of the output statement. The truth table for a three input OR gate is shown below.

INPUTS			OUTPUT
A	B	C	Z = A+B+C
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

OR

The table shows that the output statement is true (1) for all combinations where at least one input statement is true, and is false (0) only for the single combination where all three inputs are false.

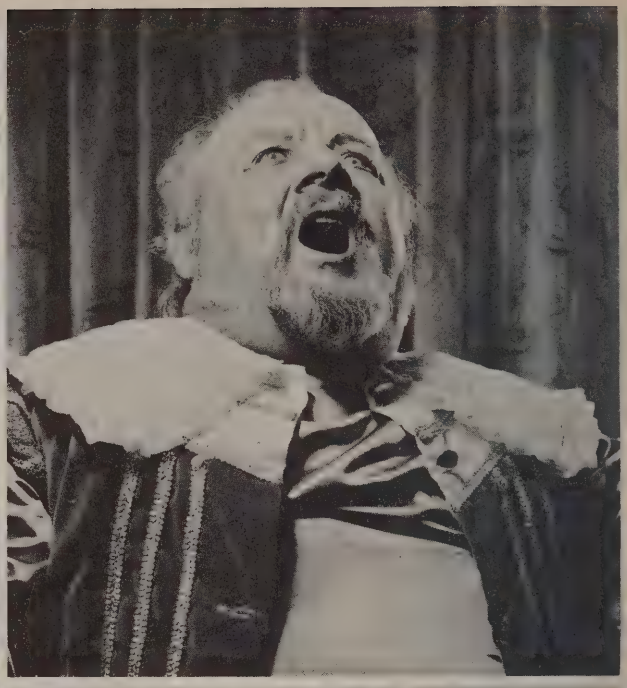
Again, the OR operation is not restricted to the three input statements shown, but may involve any number of inputs greater than one.

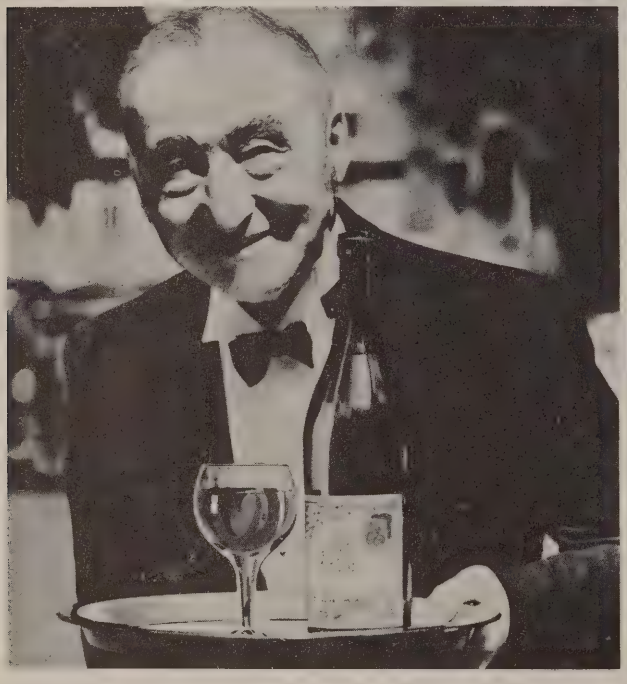
The third of the elementary logical operations is the NOT operation, which involves simply the negation, contradiction or complementation of an input statement. As explained earlier, the complement of a statement is simply another statement which is always true when the first statement is false, and vice-versa. Note that the NOT operation **as such** involves a single input statement only.

The truth table definition of the NOT operation is shown below, as before using a "bar" above the symbol for the input statement A to indicate its complementary statement not-A:

INPUT A	OUTPUT $\bar{A}$
0	1
1	0

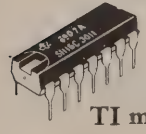
The usual logical symbols for the NOT operation are shown overleaf.





Quality, range, service, availability.

TI distributors have the complete cast for your performance.

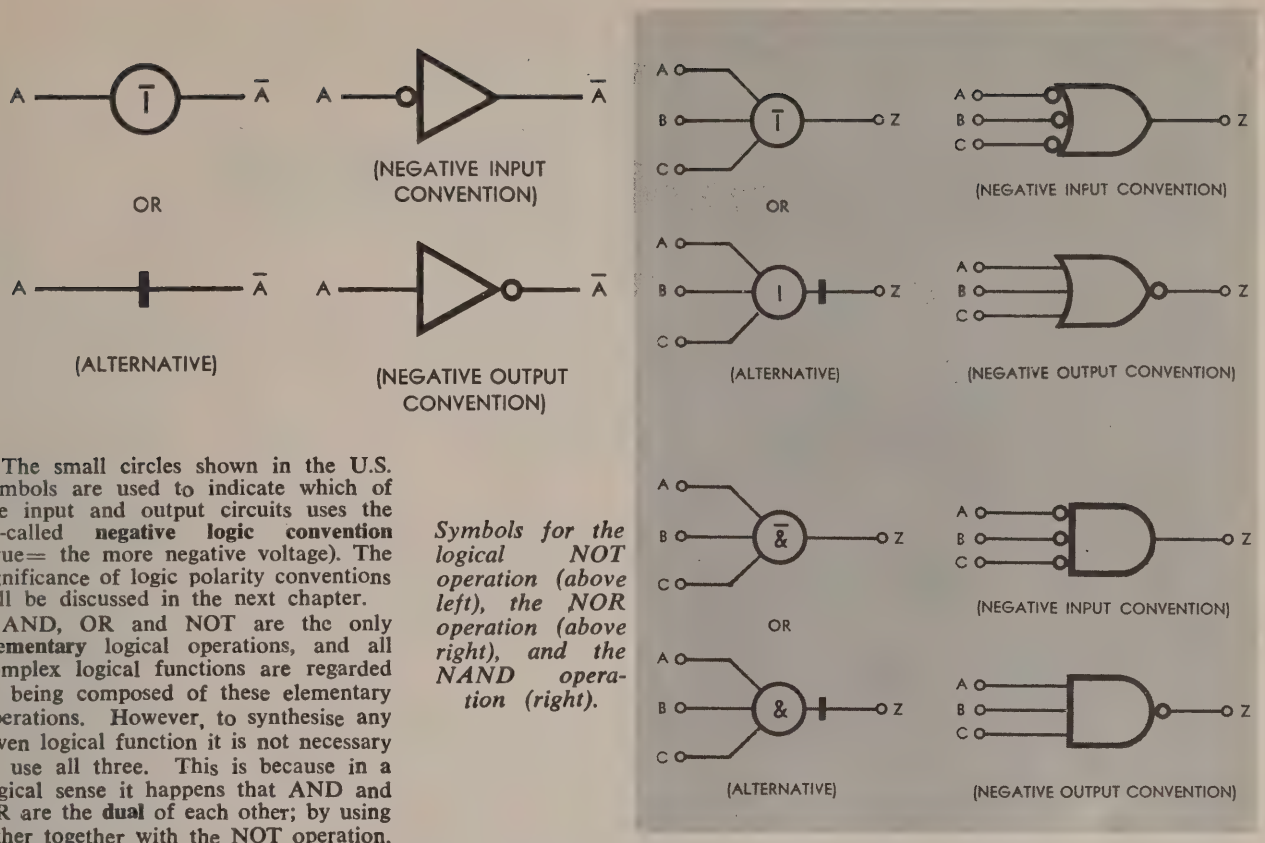


TI make the great performers

Under your direction TI will supply the products. Remember, they've been innovators since the very first performance. They've the stars for all major character roles including TTL, DTL, MOS, as well as FET's, and many discretes... all priced at the right rates.

Plan for a big circuit season... now. Contact TI for a rundown on the leading players. Phone bookings can be made from:
NSW: Electrobit Company 67 4611.
VIC: A. J. Ferguson (Melbourne) Pty. Ltd. 34 9451, Radio Parts Pty. Ltd. 30 1251. **SA:** A. J. Ferguson (Adelaide) Pty. Ltd. 23 1922.
WA: Precision Electronics 81 1533.
QLD: Douglas Electronics 4 9538.

TEXAS INSTRUMENTS
AUSTRALIA LIMITED
Head Office:
Philip Highway, Elizabeth, S A 5112. Telephone 55 2914.
Sydney Office: Room 5, Rural Bank Building, Burwood, NSW 2134. Telephone 74 1859



Symbols for the logical NOT operation (above left), the NOR operation (above right), and the NAND operation (right).

The small circles shown in the U.S. symbols are used to indicate which of the input and output circuits uses the so-called **negative logic convention** (true= the more negative voltage). The significance of logic polarity conventions will be discussed in the next chapter.

AND, OR and NOT are the only **elementary** logical operations, and all complex logical functions are regarded as being composed of these elementary operations. However, to synthesise any given logical function it is not necessary to use all three. This is because in a logical sense it happens that AND and OR are the **dual** of each other; by using either together with the NOT operation, the other can be synthetised.

Hence any logical function whatever can be synthesised using sufficient numbers of only two of the elementary operations—either AND and NOT, or OR and NOT.

Because of this it has been found convenient in practice to define two **further** "basic" logical operations which are in fact **complex** or "combination" elements consisting in each case of two of the elementary operations. These are the NOR and NAND operations, which prove particularly suitable for use as the basis of practical logic elements or "modules"—the basic "building blocks" of practical logic networks.

The NOR operation (also called the "Peirce arrow function" after its originator, C. S. Peirce) is a combination of the elementary OR and NOT operations. In other words, it is an OR operation in which the output statement is complemented. The term "NOR" is simply a contraction of NOT-OR.

The NOR operation may be represented by the relationship

$$\overline{A + B + C} = Z \qquad \ldots(4)$$

and may be defined by the truth table:

NOR

INPUTS			OUTPUT
A	B	C	$Z = \overline{A+B+C}$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

As may be seen, the output statement of a NOR element is true only when all three input statements are false, and is false provided that at least one of the inputs is true.

The usual logical symbols used for the NOR element are shown at above right.

The NAND operation (also called the "Sheffer stroke function" in honour of its originator, H. M. Sheffer) is somewhat similar to the NOR operation, but is a combination of the elementary AND and NOT operations; an AND operation in which the output statement is complemented. The term "NAND" is again simply a contraction of NOT-AND.

The NAND operation may be represented by the relationship

$$\overline{A.B.C} = Z \qquad \ldots(5)$$

and may be defined by the truth table:

NAND

INPUTS			OUTPUT
A	B	C	$Z = \overline{A.B.C}$
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

As the truth table demonstrates, the output of a NAND element is true provided that at least one of the input statements is false, and is false only when all three input statements are true.

The logical symbols used for the NAND operations are shown above.

To synthesise complex logical functions the basic logical operations described thus far are combined, using the laws of symbolic logic or "Boolean algebra." And before a synthesised logical function is turned from an abstract concept into physical components or "hardware," it is reduced to its bare essentials using techniques of **minimalisation**. A discussion of these laws and techniques forms the subject of our next chapter.

SELECTED BIBLIOGRAPHY

BENNETT, W. S., "Binary Logic—A Programmed Course," in **Product Engineering,** Jan. 8, 1962—Oct. 14, 1963. (A series of 8 articles using a "teaching machine" approach.)

BOOLE, G., **An Investigation Of The Laws Of Thought,** 1854. Reprinted by Dover Publications Inc., New York.

GARDNER, M., **Logic Machines and Diagrams,** 1958. McGraw-Hill Book Co., Inc., New York.

HENLY, H. R., "Logic Without Tears," in **Wireless World,** Vol. 71, No. 1 (January, 1965).

HILTON, A. M., **Logic, Computing Machines and Automation,** 1964. World Publishing Co., Cleveland, Ohio.

KENCH, E. J., (Ed.), **Integrated Logic Circuit Applications,** 1967. Mullard Limited, London.

LEDLEY, R. S., **Digital Computer And Control Engineering,** 1960. McGraw-Hill Book Co. Inc., New York (Esp. part 3).

SCOTT, N. R., **Analogue and Digital Computer Technology,** 1960. McGraw-Hill Book Co., Inc., New York.

LOGIC SYMBOLOGY, published by the Dymec Division of Hewlett-Packard. Available by request on company letterhead from Hewlett-Packard Australia Pty. Ltd., 22-26 Weir Street, Glen Iris, Victoria. 3146.

Chapter 2

From EQUATIONS to "HARDWARE"

Logic convention and its implications—duality and deMorgan's theorem—the laws of expansion—the remaining laws of Boolean algebra—minimalisation techniques—the Karnaugh map—practical logic circuits logic modules—inter-connection considerations—logic polarity and its manipulation—some simple RTL logic circuits—a demonstrator panel.

It was noted in the first chapter that from a logical point of view the "input" and "output" quantities associated with a circuit element or configuration are **statements** whose truth or falsity the circuit conditions are taken to represent, and that the logical-electrical equivalence or **logic convention** employed for this is an arbitrary one which may be assigned and varied as required. Let us now consider an important implication of this arbitrary nature of the logic convention.

Figure 1 shows the same simple lamp circuit which we considered in chapter 1. It may be remembered that when we took the **closure** of the three switches to represent the **truth** of three statements A, B and C, then the **lighting** of the lamp could be taken to represent the **truth** of a statement Z which was logically equivalent to the situation where statements A, B and C were all true together. In other words, when this particular logic convention was assigned the circuit operation was seen to represent the logical AND operation.

Consider now what happens if we assign to the same circuit a different logic convention. For example, if we use the **open** condition of the switches to represent the truth of their respective statements A, B and C, and the **extinguished** (zero current) condition of the lamp to represent the truth of the output statement Z. Will the circuit operation still represent the logical AND operation?

Fairly obviously, it will not. This time the output Z will be **true** (lamp extinguished) as long as **any one** of the inputs is true (switch open); it will be false (lamp alight) only when **all three** inputs are false (switches all closed). In other words the circuit will now be performing the OR operation.

Now consider what will happen if we again alter the logic convention; this time to the case where we interpret the **closure** of the switches to represent the truth of the input quantities but with the **extinguished** state of the lamp to represent the truth of the output statement. A little reflection will show that this time the circuit will be performing the logical NAND operation—the output is false only when all three inputs are true.

And finally, if we adopt a convention where the **open** condition of the switches represents the truth of the input statements while the **lighted** condition of the lamp represents the truth of the output, we shall find that the circuit will be performing the NOR operation. We can make it perform the NOT operation as a "special case" with this convention by using only one of the input switches and leaving the other two permanently closed to represent continuously false statements.

It should be fairly clear from the foregoing that the interpretation placed upon the operation of a logic circuit element or configuration depends largely upon the logic convention assigned to it. Any given logic element may thus be arranged to perform any of the basic logical operations simply by assigning to it the appropriate logic convention. As we shall see later in this chapter this fundamental fact of circuit logic can be used to great advantage in the reduction of logic circuits to their barest essentials.

There is particular significance in the fact that an AND circuit element can be effectively "converted" into an OR element, or vice-versa, by reversing the logic convention, and in the fact that the NAND and NOR elements are similarly inter-related. For it is an important logical truth that in each case these operations are the converse or dual of one another.

They are virtually the opposite sides of the same logical "coin," as it were; a fact which should be evident from a consideration of figure 1. The complement of an AND operation on a given set of quantities can be seen to be logically equivalent to an OR operation on their individual complements.

This relationship between AND and OR and NAND and NOR is normally called either the "law of duality" or **"deMorgan's theorem,"** the latter in honour of Boole's contemporary. It forms one of the axioms of symbolic logic or **"Boolean algebra,"** and may be symbolised concisely by the two expressions

$$\overline{(A + B)} = \overline{A} \cdot \overline{B} \qquad \ldots(1)$$

$$\overline{(A \cdot B)} = \overline{A} + \overline{B} \qquad \ldots(2)$$

In words these state the two aspects of the theorem as "the complement of a logical sum ('NOT'[A or B]') is equivalent to the logical product of the complements of the individual terms ('not-A and not-B')," and "the complement of a logical product ('NOT [A and B]') is equivalent to the logical sum of the individual complements ('not-A or not-B')."

Truth tables may be used to show that deMorgan's theorem is in fact a logical axiom and true in all cases. The first aspect may be shown to be universally true by the table

ORIGINAL STATEMENTS		COMPLEMENTS		SYNTHESISED STATEMENTS		
A	B	$\overline{A}$	$\overline{B}$	A+B	$\overline{A+B}$	$\overline{A}.\overline{B}$
0	0	1	1	0	1	1
0	1	1	0	1	0	0
1	0	0	1	1	0	0
1	1	0	0	1	0	0

Here the first two columns represent the elementary statements, with the four possible combinations of truth and falsity of these corresponding to the four horizontal rows (0 = false, 1 = true). The third and fourth columns show the complements of the elementary statements, while the three final columns show the synthesised statements.

In the fifth column is the statement "A or B" and in the sixth its complement "NOT (A or B)." In the seventh column, for comparison, is the statement "not-A and not-B."

As may be seen, the "truth values" of the last two synthesised statements are identical. The statements are always true together and false together, so that they are logically equivalent to one another.

The second aspect of deMorgan's

The logical interpretation placed upon the operation of a circuit depends upon the adopted logic convention, as this diagram is used to demonstrate.

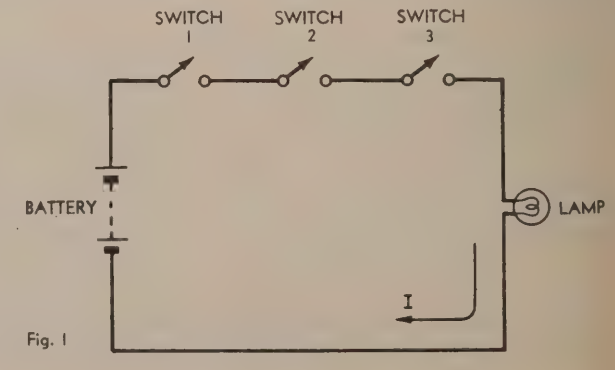
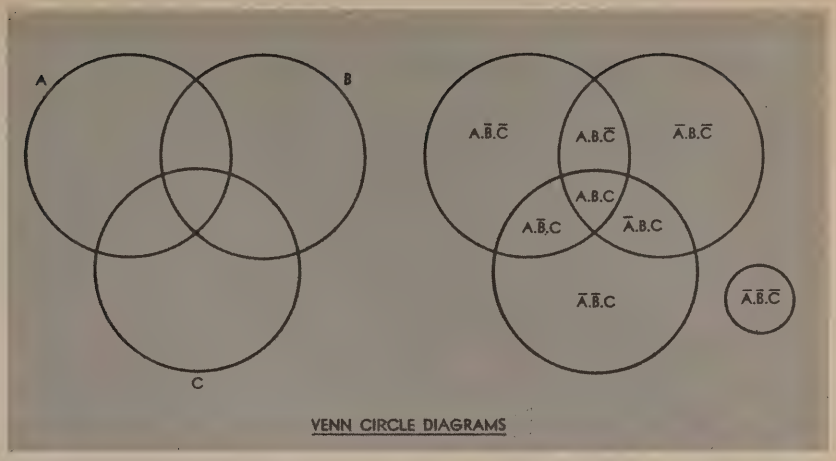


Fig. 1



VENN CIRCLE DIAGRAMS

Topological diagrams provide a particularly concise method of representing complex logical functions. Shown here are examples of the circle diagrams developed around 1890 by the Cambridge logician John Venn. With a compartment for every possible term combination, they were the predecessors of the Karnaugh maps discussed in the text.

theorem can be seen to be equally universal, by a similar truth table:

A	B	$\bar{A}$	$\bar{B}$	A.B	$\overline{A.B}$	$\bar{A}+\bar{B}$
0	0	1	1	0	1	1
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	1	0	0

Here again the truth values of the last two columns are identical, showing in this case that the statement "NOT (A and B)" is logically equivalent to the statement "not-A or not-B."

Before we pass on to consider a second axiom of symbolic logic note carefully that two statements may be regarded as logically equivalent if they are always true together and always false together.

Consider now the situation represented by the following simple logic diagram:

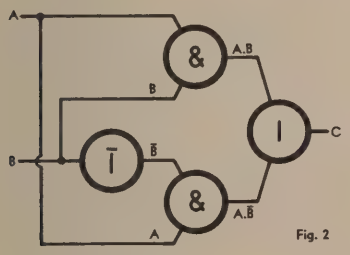


Fig. 2

The logical relation representing this situation may be written as

$$(A \cdot B) + (A \cdot \bar{B}) = C \qquad \ldots(3)$$

If you look at either the diagram or the symbolic expression for a moment it should become evident that the elementary term "B" is actually redundant. This is simply because the output can be true both when B is true and also when B is false and its complement true.

In fact the output C is solely dependent upon the truth of the elementary input statement A. Not only this, but the output is actually the logical equivalent of A — when A is true the output is true, and when A is false the output is false. Hence we may re-write the symbolic expression describing this logical situation as

$$(A \cdot B) + (A \cdot \bar{B}) = A \qquad \ldots(4)$$

Reference to the corresponding truth table shows that this is again a universal logical truth:

A	B	$\bar{B}$	A.B	A.$\bar{B}$	(A.B)+(A.$\bar{B}$)
0	0	1	0	0	0
0	1	0	0	0	0
1	0	1	0	1	1
1	1	0	1	0	1

The first and last columns have identical truth values, showing that the statement "(A and B) or (A and not-B)" is always logically equivalent to statement A. Hence the four logic operations shown in figure 2 may be replaced by a simple "connection" from the output to input A, **without any change in logical function.**

Expression (4) symbolises one of a pair of axioms of symbolic logic known as the **"laws of expansion."** The second of the pair may be expressed symbolically as:

$$(A + B) \cdot (A + \bar{B}) = A \qquad \ldots(5)$$

This expresses a situation very similar to that which we have just considered; figure 3 shows the corresponding logic diagram. Again it may be seen that the elementary term "B" is redundant, and that the four operations could be replaced by a simple "connection" without any change in the logical function. The reader himself may care to verify that this is universally true, by drawing up the appropriate truth table.

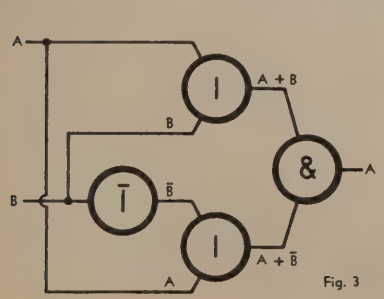
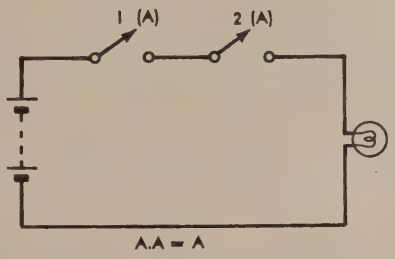


Fig. 3



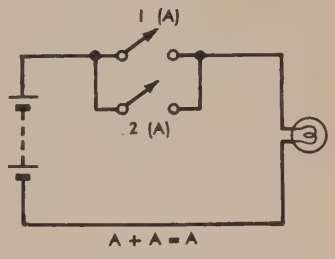


Figure 4: In each of the diagrams above, because the switches represent identical terms they must be either both open or both closed. Each pair of switches is therefore equivalent to a single switch, illustrating the logical "laws of tautology."

Note that in discussing the laws of expansion we have introduced the concept of logical redundancy. A statement "term" forming part of a complex statement may be regarded as logically redundant if it may be either true or false without affecting the truth value of the complex statement.

No doubt at this stage the more astute reader will already have begun to foresee the importance of deMorgan's theorem and the laws of expansion from the point of view of logic circuit design. However before we pass on to consider such practical matters it will be worthwhile to examine briefly the remaining axioms of symbolic logic.

The **"laws of tautology"** point out that the logical sum (OR) or the logical product (AND) of two identical terms is logically equivalent to either input:

$$A + A = A \qquad \ldots(6)$$

$$A \cdot A = A \qquad \ldots(7)$$

In other words, there is no need to add a statement to itself ("A or A"), or to "square" a statement ("A and A"). In both cases the output is simply the same as the input.

An example demonstrates the truth of these laws quite clearly: If "A" stands for the statement "It is raining," there is no point in saying either "It is raining OR it is raining" or "It is raining AND it is raining." In both cases we are still from a logical point of view say-

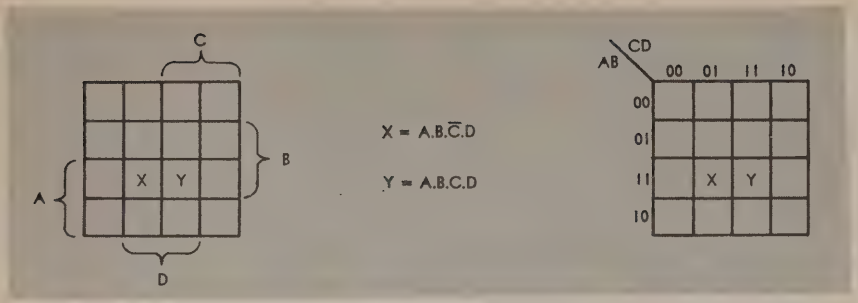


Figure 5: Karnaugh map diagrams for logical functions having four component terms; there are 16 cells corresponding to all possible truth value combinations of the terms. The two methods of representation shown are equivalent.

ing only "It is raining" and needlessly repeating ourselves.

Figure 4 shows simple lamp circuits which also demonstrate the truth of the laws of tautology. In each case since both switches are used to represent statement A, they must be operated together; hence in both cases they become equivalent to a single switch.

For brevity, the remaining axioms will be defined as symbolic expressions only. The reader may care to verify them for himself by drawing appropriate logic diagrams, truth tables and simple switch circuits.

The **"laws of commutation"** point out that the order of terms makes no difference in either the AND or OR operations:

$$A + B = B + A \quad \ldots(8)$$

$$A \cdot B = B \cdot A \quad \ldots(9)$$

i.e., "A or B" is exactly the same as "B or A," and so on.

The **"laws of association"** show that the order of carrying out consecutive operations of the same type is unimportant:

$$(A+B)+C=A+(B+C) \quad \ldots(10)$$

$$(A \cdot B) \cdot C = A \cdot (B \cdot C) \quad \ldots(11)$$

The **"laws of distribution"** show that the AND and OR operations are distributive over each other:

$$A+(B \cdot C) = (A+B) \cdot (A+C) \quad \ldots(12)$$

$$A \cdot (B+C) = (A \cdot B)+(A \cdot C) \quad \ldots(13)$$

The **"laws of absorption"** show that either a sum (OR) or a product (AND) in which the terms contain a common statement is equivalent to that common term:

$$A + (A \cdot B) = A \quad \ldots(14)$$

$$A \cdot (A + B) = A \quad \ldots(15)$$

The **"law of double negation"** is simply a formal statement of the truism that a statement is the complement of its complement, or that a double negation restores a statement to its original value:

$$\text{not } (\overline{A}) = A \quad \ldots(16)$$

The three final axioms make use of the fact that by definition the symbol "1" may be taken to represent a statement which is always true, and that similarly the symbol "0" may be taken to represent a statement which is always false.

The **"laws of the universe class"** show that the output of a sum (OR) in which one term is always true is also always true, and that the output of a product (AND) in which one term is always true is simply equivalent to the other term(s):

$$A + 1 = 1 \quad \ldots(17)$$

$$A \cdot 1 = A \quad \ldots(18)$$

Conversely, the **"laws of the null class"** show that the output of a sum (OR) in which one term is always false is simply equivalent to the other terms(s) of the sum, while the output of a product (AND) in which one term is always false is itself always false.

$$A + 0 = A \quad \ldots(19)$$

$$A \cdot 0 = 0 \quad \ldots(20)$$

And finally the **"laws of complementation"** show that the sum (OR) of a term and its complement must always be true, and that the product (AND) of a term and its complement must always be false:

$$A + \overline{A} = 1 \quad \ldots(21)$$

$$A \cdot \overline{A} = 0 \quad \ldots(22)$$

In other words, either a statement is true or its complement must be true; while a statement and its complement can never be true at the same time.

Let us now turn to consider what is involved when it is desired to produce a practical logic network capable of performing a given complex logical function. This is often termed **realisation** or "implementation" of the logic in terms of electronic "hardware."

In general, the aim in producing practical logic circuitry is to make the circuitry as efficient as possible — not only electrically, but logically and economically as well; although quite often all three aspects of efficiency merge. The goal is to produce the simplest and cheapest circuit capable of performing the required logical functions reliably and with the minimum power consumption.

As one might perhaps expect, both logical and electronic methods are used in this quest for efficiency. Generally the technique is to work first at the logical level, distilling the required logical function to arrive at its simplest and most essential form, and then to move to the electronic level in an effort to arrive at the simplest and most efficient circuit capable of performing the essential logic. However the two spheres are complementary rather than independent, and often it proves worthwhile to modify the essential logic to take advantage of certain characteristics of the particular devices or circuitry to be used.

As we noted at the conclusion of the first chapter, the logical aspect of the task is termed **minimalisation.** Logically this may be defined as a process of finding the "prime implicants," where the prime implicants are those minimum logical terms which are both necessary and sufficient to completely perform the required complex logical function.

There are a number of approaches used in the effort to find the prime implicants. One approach is to simply work with pen and paper in time-honoured "cut and try" fashion with the logic diagram, re-arranging it and modifying the logical configurations until no further simplification can be envisaged. A somewhat more reliable approach is to manipulate the symbolic logic expression for the required function, using the various axioms of "Boolean algebra" which we have examined earlier in this chapter. The concise nature of the final symbolic expression produced makes it somewhat easier to verify that one has in fact found the prime implicants.

Unfortunately the expression describing the required logic function usually becomes somewhat unwieldy when there are more than about four input terms or "variables." As a result it often proves useful to employ graphical and topological methods. Of these probably the most popular is the Karnaugh map method, developed from the earlier topological map systems of Veitch, Venn and Euler.

The Karnaugh map is a multi-compartment matrix diagram in which each compartment or "cell" corresponds to a specific possible combination of the input variables concerned.

Karnaugh maps for the topological representation of two and five-variable logical functions.

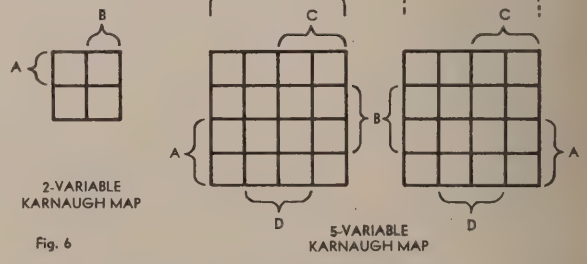


Fig. 6

A Karnaugh map for functions involving four variables has 16 cells, corresponding to the possible combinations of truth or falsity of the four variables, and the cells are usually arranged as shown in figure 5. Each of the four variables A, B, C, and D are regarded as having the value "true" for the cells indicated by the appropriate brackets, and the value "false" for the remaining cells.

To illustrate the cell coding, cell "X"

in the figure represents the variable combination expressed by

$$X = A.B.\overline{C}.D$$

and the adjacent cell "Y" represents the combination given by

$$Y = A.B.C.D$$

From a consideration of figure 5 it should become evident from the coding of the cells results in the property whereby each cell differs from those immediately adjacent to it in the value of only one variable. For example the cells marked X and Y differ only with respect to the value of the variable C, while cell X differs from that immediately above it in the diagram only with respect to the value of variable B. This property permits cell grouping, as will be seen in a moment.

Figure 6 shows examples of Karnaugh maps for other numbers of variables. The two-variable map is shown purely for illustration; in fact a map would never be required where only two variables were involved. Note that for five variables it proves convenient to draw the map as two matrices, one corresponding to each value of the fifth variable "E".

By putting a "1" or a "0" in each cell of the Karnaugh map to signify whether the **output** of the given complex logical function is required to be true or false (respectively), the map becomes a highly concise and readily conceived symbolic representation of the function concerned. As a result it becomes relatively easy to separate the prime implicants from any redundant terms, and to see the various possibilities for synthesis of a minimal expression of the function from the original expanded expression.

This is perhaps best shown by an example. The first step in using the Karnaugh map method is to write down, in full, the required logic function. In general this means stating all the input variable combinations for which the output is required to be true. Thus in a particular application we may require the output Z to be true for the following combinations of four input variables A, B, C, and D:

$$\begin{aligned} Z = {} & A.B.C.D + A.\overline{B}.C.D \\ & + A.B.\overline{C}.D + \overline{A}.B.C.D \\ & + \overline{A}.B.C.\overline{D} + \overline{A}.B.\overline{C}.D \\ & + \overline{A}.B.\overline{C.D} + \overline{A.B.C}.D \end{aligned} \quad \ldots(23)$$

In words, the output Z is here required to be true whenever there occurs any one or more of the input variable combinations "(A and B and C and D)," or "(A and not-B and C and D)," or "(A and B and not-C and D)", etc. . . .

The next step is to enter this information into a Karnaugh map appropriate for the number of variables involved—here four. This is done by placing a "1" in each cell corresponding to a term in expression (23). The remaining squares are marked according to whether or not the output is required to be false for the combination concerned.

There will be many applications in which the output will be definitely required to be **false** for all input combinations other than those defined by the given logic expression, and in such cases a "0" will be placed in every cell in which there has not been placed a "1." However there will be certain applications in which it will not be **essential** that the output be false for all undefined combinations, and in these latter cases it is usual to place an "X" in those cells where the output could be either **true or false** without significantly affecting the normal logical function. A common situation in which such "don't care" combinations occur is where it is known beforehand that certain of the undefined combinations can never occur.

Figure 7 shows a four-variable map with the cells marked for the function of expression (23), assuming that there will be no "don't care" combinations in the application concerned.

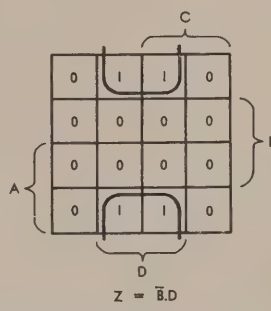
Having derived the Karnaugh map of the function concerned the next step is to attempt to find the most concise way of describing the total function as a set of small sub-functions. This is done by attempting to group all the "1" cells into a minimum number of small groups in which one or more variables are redundant.

At this stage it may be worthwhile to re-state the definition of logical redundancy given earlier in this chapter: namely that a statement term (variable) that forms part of a complex statement (function) may be regarded as logically redundant if it may be either true or false without affecting the truth value of the complex statement (function). Hence if a problem required that an output S were to be true only for the two combinations of A, B, C, and D represented by cells "X" and "Y" of figure 5, this would mean that as far as S were concerned variable C could be ignored as redundant—it can clearly be either true or false without affecting the truth of S. One could thus define the function of the required logic circuit as "S = A.B.D"

The idea, then, is to attempt to group the "1" cells of the map of figure 7 so that as a whole they can be described as a minimum number of small groups.

Generally the procedure is to first isolate any cells which cannot be grouped with any others. Then "loops of two" groups are made around those cells which will so group yet will not group into larger loops; then "loops of four" groups are made around those which cannot be grouped into still larger loops, and so on. Note that it is only possible to group cells in loops enclosing numbers of cells given by the positive integral powers of two—2, 4, 8, 16, etc. This follows from the fact that cell grouping relies upon variable redundancy.

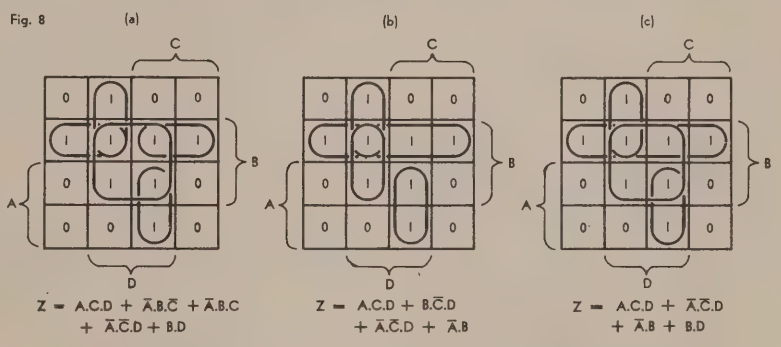
In theory the Karnaugh map has no "sides"; it is a topological surface equivalent to the surface of a sphere. Hence cells on opposite edges of the matrix representation are effectively adjacent to each other, and may be grouped together thus:



Here the loop encloses four cells which as a group correspond to a function in which both A and C are redundant, B is always false and D is always true, i.e.,

$$Z = \overline{B}.D$$

In our example of figure 7 this type of loop does not prove relevant. However as occurs in many cases there happens to be a number of possible "correct" answers, and three of these are shown in figure 8 together with their corresponding prime implicants.



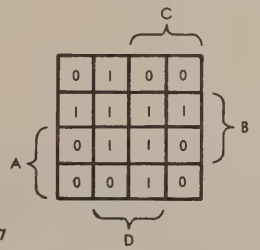


Fig. 7

From a purely logical viewpoint the grouping marked (c) would probably be regarded as giving the most basic expression for the original logical function, because as may be seen it reduces the function to only four terms involving the smallest number of variables and their complements. There are two loops of two cells each giving terms having three variables, and two loops of four cells each giving terms having only two variables. Not only this but only variables A and C are required in both original and complement form.

The logic diagram which would result from this grouping is shown for illustration in figure 9. It can be seen to require only four AND gates, a single OR gate and two inverters (NOT elements) — seven elements in all.

While in this case the grouping in (c) seems to give the greatest simplification of the function, there will be many cases in which there will be a

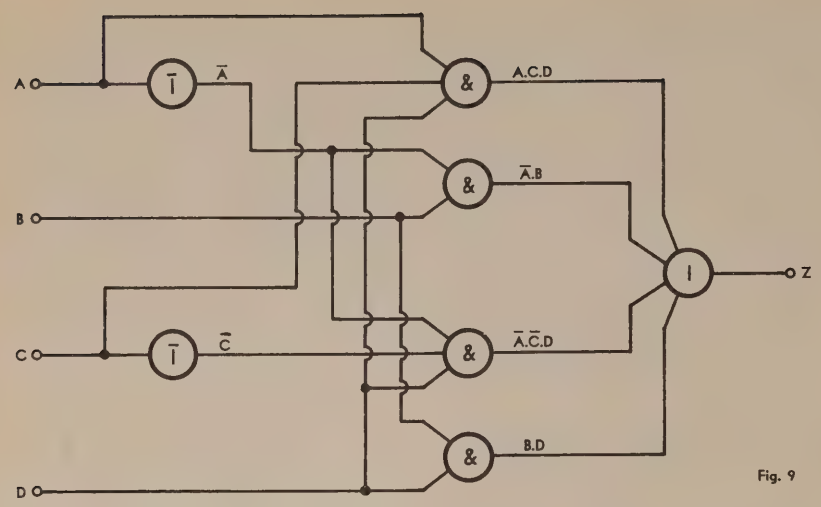


Fig. 9

number of answers equally basic from a purely logical viewpoint. In such cases the final choice between the various alternatives will be influenced by non-logical factors such as the type of circuit blocks to be used, their input and output loading limitations, layout, and the convenience of obtaining complements of input variables if these prove to be required.

It may be seen from this brief introduction that the Karnaugh map method allows one to determine fairly easily the various possible simplifications of the required complex logic function. This allows the selection of the most economical and efficient of these before it is attempted to realise the logic in terms of practical circuitry.

From the logical side of the realisation process let us now turn to consider the more practical aspects. We will first look at the actual circuitry involved in performing the elementary operations AND, OR, NOT, NOR and NAND — the circuit blocks or "gates."

There are a great many different types of devices and circuits used to perform the elementary logic operations, and it will not be possible here to describe even a majority of these in any great detail. However, the paragraphs which follow describe the more common types of circuits and devices in current use.

WIRED LOGIC: This is the simplest logic circuitry of all, consisting of nothing more than direct connections between input(s) and output. Depending upon the logic conventions in use and the rest of the circuitry, such simple direct connections can effectively perform any of the elementary logical operations. However they can only be used in conjunction with other types of logic circuitry, and then sparingly, because a wired connection usually precludes additional connections to the same input sources and may often introduce loading problems. Wired operations are usually called "wired-AND," "wired-OR," and so on.

SWITCH AND RELAY LOGIC: Uses mechanical switch contacts to perform the logic, operated manually or magnetically or by motion of a mechanical part of the system. Because of the shortcomings of mechanical contacts, used mainly for slow-speed logic and "front panel" control of equipment operation.

DIODE LOGIC: A simple system of electronic logic employing semiconductor diodes and resistors to perform the logic, with either transistor or integrated circuit (IC) amplifiers to maintain signal levels. Can attain very high speeds of operation. Often abbreviated to "DL".

TUNNEL DIODE LOGIC: Not much more complex than ordinary diode logic, but employs the unique characteristics of the tunnel diode to attain extremely high operating speeds. Often abbreviated to "TDL".

NEON LOGIC: Uses neon lamps or three-element neon trigger tubes to perform the logic, again in conjunction with resistors. Offers the advantage of economy where operating speeds can be quite low; not suitable for high speeds. Abbreviated to "NL".

OPTO-ELECTRONIC LOGIC: Uses either neon, incandescent or semiconductor light sources and photo-sensitive elements to perform the logic. Depending on the light sources and photo-elements used, can offer a variety of operating speeds for a relatively low cost. Also has the feature of electrical isolation between inputs and outputs, often an important advantage in applications such as encoding and decoding.

CORE DIODE LOGIC: Here logic is performed by semiconductor diodes and small (usually toroidal) magnetic cores made from ferrite or a similar material. For moderate speeds and moderate power levels. Usually abbreviated to "CDL".

POWER LOGIC: Usually employs thyristors (SCRs), 4-layer diodes or silicon controlled switches to perform the logic, which in this case involves high currents and/or voltages.

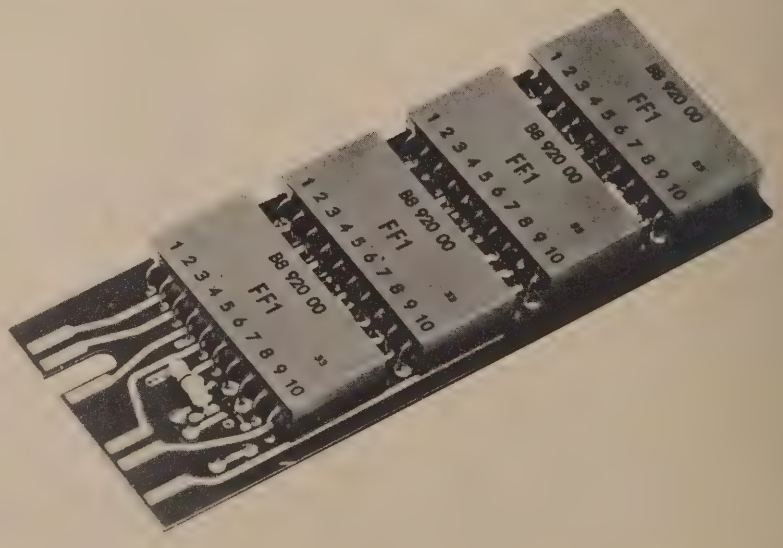
DIRECT COUPLED TRANSISTOR LOGIC: Uses transistors switching between the saturated ("bottomed") and cut off states to perform the logic. Depends largely upon the use of transistors with carefully controlled parameters. Economical but tends to be susceptible to noise because of the low voltage levels employed; moderately fast. Abbreviated to "DCTL".

RESISTOR TRANSISTOR LOGIC: Perhaps the most straightforward and economical of the electronic logics where only moderate speeds of operation are required. The resistors perform most of the logic, with the transistors as before performing inversion and amplification. The use of silicon transistors gives good temperature stability, circuit simplicity and reliability. Abbreviated to "RTL," or "RCTL" where capacitors are used to give a marginal increase in operating speed.

DIODE TRANSISTOR LOGIC: Virtually an extension of simple diode logic, with diodes to perform the majority of the logic and transistors for inversion and maintaining voltage levels. Capable of high speed operation at moderate cost. Abbreviated to "DTL".

TRANSISTOR-TRANSISTOR LOGIC: A development from DTL, in which the logic is usually performed by special multi-emitter transistors rather than diodes. Lower circuit impedances are attainable, increasing the operating speeds. Usually abbreviated to "TTL".

CURRENT MODE LOGIC: Transistors again perform the logic, but circuit configurations are used which prevent the transistors from saturating. This reduces charge-storage effects to a very low level and permits extremely high



Four of the Mullard "Series 1" logic modules mounted on a plug-in printed wiring board. Each module consists of "discrete" components mounted on a printed wiring board and encapsulated in a synthetic resin case. (Courtesy Mullard-Australia Pty. Ltd.)

speed operation. Has a high noise immunity despite fairly low voltage levels. Abbreviated to "CML".

Space limitations prevent a more complete or more detailed listing here of the types of logic circuitry currently available; however it is hoped that the foregoing will give the reader a glimpse of the variety of circuits available. Some of the more common types are shown in basic form in figure 10.

Note that in realising logical functions in terms of circuitry one is not restricted to consistent use of a single type of logic circuit element; in fact it is quite common for designers to use a number of different types of circuitry in combination, in an effort to achieve an optimum performance/cost ratio. With a few exceptions most types of logic circuitry may be made electrically compatible with one another.

Although logic circuitry can be — and was originally — wired up "normally" as complete and homogenous systems, it has generally been found more efficient to assemble it using a "module" or "building block" approach. This is simply because the complex logical functions to be realised already consist of synthetic configurations of the basic logical operations AND, OR, NOT, NAND and NOR, and can therefore be realised most efficiently by appropriately interconnecting groups of modules which can perform these basic operations.

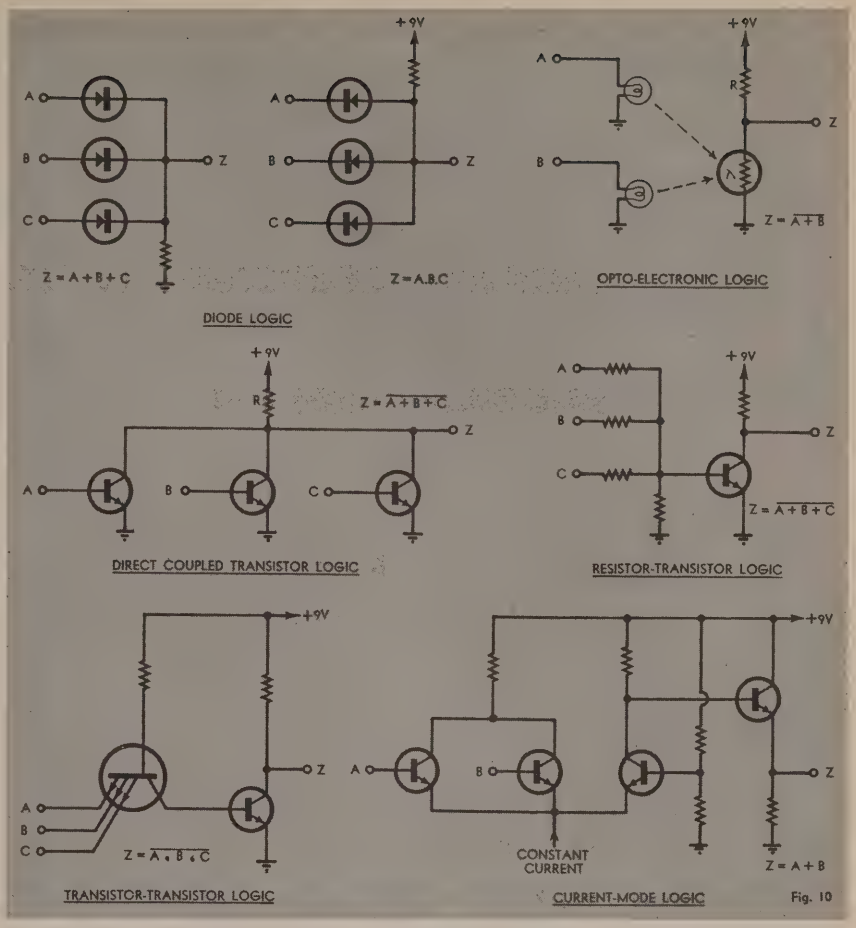
Because of the complex nature of the NAND and NOR operations which makes it possible to use either alone to synthesise any logical function, it is common to use these operations as the basis for commercial logic modules.

Commercial modules are of two general types. One type is fabricated by more or less conventional techniques using discrete components, while the other is fabricated as an integrated microcircuit. The latter type are gradually assuming greater and greater importance as a result of reduced manufacturing costs, higher reliability and a marked reduction in bulk; it is likely that before long the discrete component type will be obsolescent.

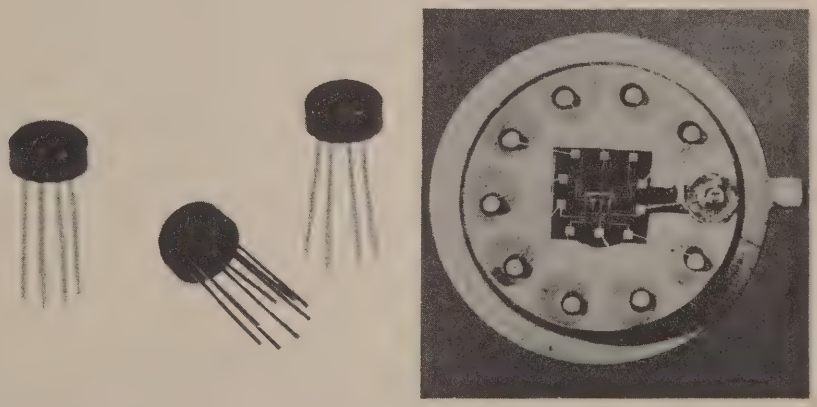
Discrete component modules are currently used where extreme compactness is not required. They consist of the appropriate resistors, transistors, diodes and capacitors wired up as an assembly and usually encapsulated or "potted" into a protective epoxy resin block.

Typical blocks of this type are illustrated, those shown being from the "Series 1" digital circuit block range marketed by Mullard-Australia Pty. Ltd. Operating at speeds up to 100KHz, they are intended primarily for use in industrial process control, low speed computers, test equipment, medical electronics and similar applications. Each block measures 54mM x 24mM x 10mM and weighs approximately 20gms; operation is from ± 6V.

The more recently developed **integrated microcircuits** (often abbreviated to "ICs") offer a most attractive alternative to the discrete component modules where compactness and reliability are important. There are a number of different varieties of integrated microcircuits, but most achieve a similar result whereby the majority if not the whole of the circuit elements are fabricated from a single tiny chip of semiconductor material. The chip is then encapsulated, often becoming a package no larger than many transistors.



The basic circuits shown above illustrate some of the more common varieties of logic circuitry. The logic expressions adjacent to each symbolise the appropriate function, assuming a positive logic convention.



At left are samples from the Fairchild "Micrologic" range of integrated microcircuits, a little larger than actual size. At right is a much enlarged view of the interior of a typical microcircuit.

As most of the processes involved in integrated microcircuit manufacture are eminently suitable for automatic control, the cost of the modules can accordingly be made quite low. Pictured as typical examples of this type of module are samples from the "Micrologic" range manufactured by Fairchild Australia Pty. Ltd., using their patented silicon planar process. At the time of writing epoxy encapsulated devices in this range are marketed at a small quantity rate from as low as $1.05, which compares more than favourably with discrete component modules.

It is common for manufacturers of both discrete-component and integrated microcircuit modules to rate their units in terms which stipulate the maximum number of inter-module connections permissible for reliable operation. The usual rating scheme is to specify the effective loading of each input of the module on preceding modules — the **"fan-in"** — together with the permissible output loading or **"fan-out."** Note that fan-in describes the loading effect of each input of the module itself, while fan-out prescribes the type and number of modules which may follow it.

These "loading" and "drive" factors are usually expressed not as voltages,

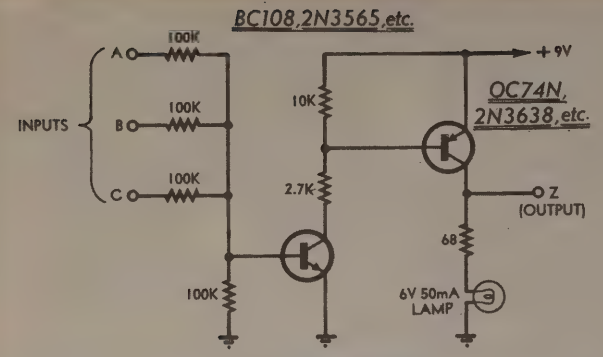


Fig. 11 RTL "OR" GATE

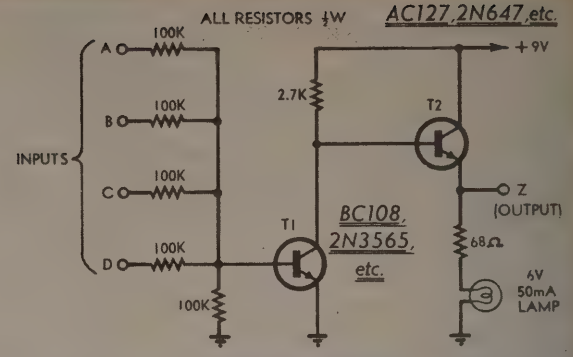
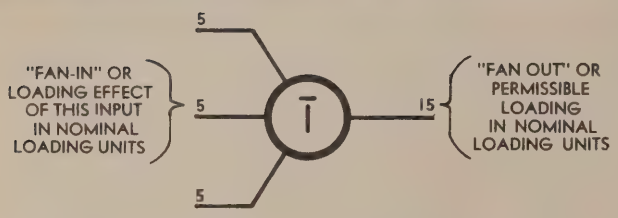


Fig. 12 RTL "NOR" GATE



Explanation of the loading terms "fan-in" and "fan-out."

currents or impedances, but in multiples of a nominal loading unit — often equal to the loading effect of a single module input. Thus the Fairchild microcircuits pictured are given fan-in and fan-out ratings which are multiples of the nominal loading of a single input of one of their low-power modules, approximately 200uA at 0.9V.

An inverter (NOT) module may accordingly be specified as having a fan-in of "6" units and a fan-out of "80," showing that its input loads a preceding module by 6 units while the output may be loaded with modules together not exceeding 80 units. It may be seen that the use of nominal loading units can simplify the procedure of interconnecting modules while ensuring that each module operates reliably.

As one would expect from the discussion of logic convention at the beginning of this chapter, most logic modules are capable of performing a variety of logical functions. However, it is usual for the manufacturer to adopt a nominal logic convention for the purpose of specifying the module performance, and this permits each module to be given a "nominal" function.

Thus far in our consideration of logic convention we have considered only switch-contact logic, in which the logic statements are represented by the open and closed states of contacts or the flow of non-flow of currents. However with wired logic and with most types of electronic logic circuitry it is more usually the case that the logical statements are represented rather by differing voltage or potential levels. Thus it is usual to talk in terms of a **logic polarity convention.**

In so-called **"positive logic"** circuitry, the true or 1 state of the circuit is represented by a **more positive** potential than the false or 0 state. With a positive supply line and negative chassis or "ground," positive logic circuitry thus employs a near-supply potential to signify "1" and a near-chassis potential to signify "0." However, a circuit with a negative supply line and positive chassis may still use positive logic, this time by using a near-chassis potential to signify "1" and a near-supply potential to signify "0."

Precisely opposite conventions are adopted with so-called **"negative logic"** circuitry: here the more negative circuit potential is used to represent the "1" state and the more positive the "0" state. Again, the logic polarity convention is quite independent of the actual power supply polarity of the circuit, and it is quite feasible to have a positive supply line yet interpret the function of the circuit in terms of negative logic.

The fact that it is possible to interpret electrical circuit operation in terms of both positive and negative polarity conventions means that very often modules can be made redundant, and omitted simply by alternating when appropriate between one convention and the other. For instance, a NOT operation can often be provided simply by reversing the polarity convention, obviating the need to use an inverter module.

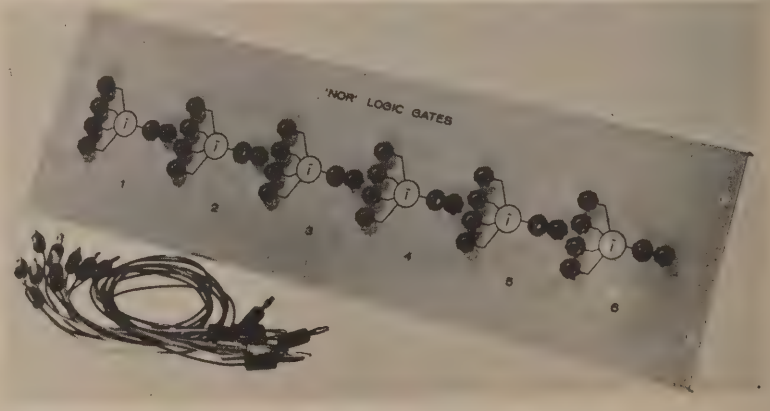
It is by no means essential that all connections to a module be interpreted using the same polarity convention. In fact it very often proves convenient to use different conventions at different terminals to a module; not only this, but in complex logic circuitry it is not unusual to have multiple conventions at single module terminals—so that one connection to a terminal may interpret the working potentials from a positive logic viewpoint, while another may interpret the potentials from the viewpoint of negative logic. Such techniques are usually referred to as **"mixed polarity logic."**

Manipulation of the logic polarity convention is an important technique available to the designer of digital circuitry, and as such forms a powerful "practical" supplement to the purely logical technique of minimalisation. Use of both techniques is virtually essential if the designer is to arrive at the most efficient and effective realisation of the required logical functions.

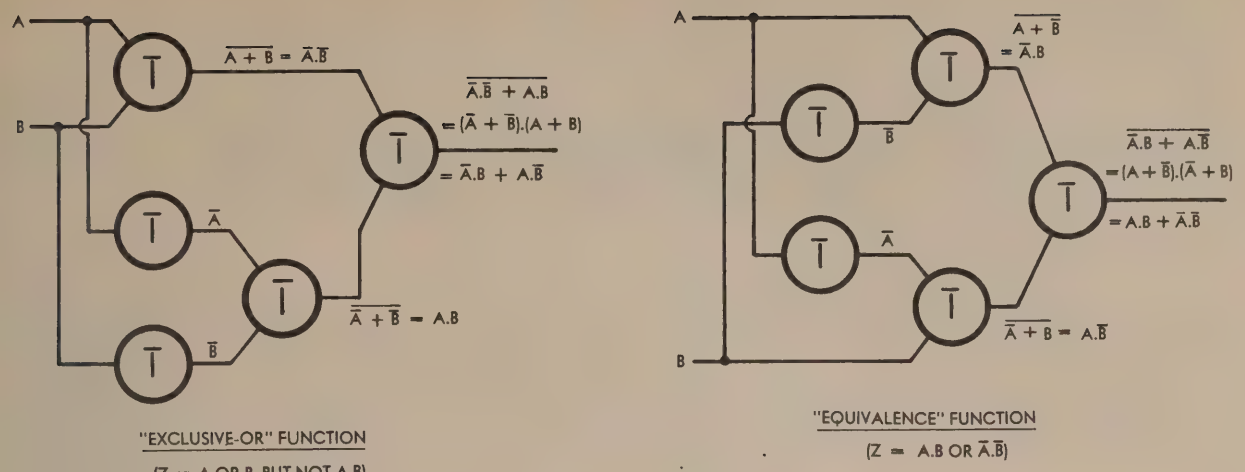
To conclude the present chapter let us look briefly at some simple logic module circuits.

The diagram of figure 11 shows a simple two transistor RTL circuit which from a **positive logic** point of view performs the OR operation. Application of a +9V input signal to any of the three input terminals A, B or C will cause the transistors to conduct, raising the potential of the output terminal Z to approximately +9V and also causing the small indicator lamp to light.

The same circuit becomes an AND gate if it is considered from a **negative logic** viewpoint. Hence the output Z will be true (chassis potential) only when all three inputs A, B and C are true also. However, note that in this case the indicator lamp will signify the false output



Although the six logic elements on this demonstration panel are nominally labelled "NOR" gates, they may in fact be arranged to perform any of the five basic logical functions. Construction of the panel is described in chapter 12; it may be constructed alone or as part of a complete digital demostrator.



EXAMPLES OF "NOR" LOGIC

Fig. 14

Two of the many logical functions which may be synthesised using the six gates of the demonstrator panel shown on the previous page. The gates can also be used for demonstrating the "flip-flop" and for binary readout.

condition rather than the true output condition.

From a "mixed logic" viewpoint the circuit has two further potential functions. If the positive convention is adopted at the inputs with the negative convention at the output ("1" = +9V at the inputs and 0V at the output), then the circuit will perform the NOR operation: the output will be true only if all three inputs are false. Conversely if the negative convention is adopted at the inputs with the positive convention at the output, the circuit will be seen to perform the NAND operation; the output will be false only when all three inputs are true.

With either of the mixed logic polarity conventions the circuit will also perform the NOT operation if all but one of the inputs are left unconnected. This follows from the fact that NOT is logically equivalent to either NOR or NAND with a single input statement.

In similar fashion the circuit of figure 12 may also be used to perform any of the elementary logical operations. Here with positive logic the circuit performs the NOR operation, with negative logic it performs the NAND operation, with positive-negative mixed logic the OR operation and with negative-positive mixed logic the AND operation. And with either positive or negative logic it becomes a NOT gate if only one input is used.

The circuits of figures 11 and 12 are quite practical and are well suited for demonstration and low speed experimental work. They offer fairly high input impedance (100K) for low fan-in, quite low output impedance for high fan-out, and inbuilt indicating lamps to indicate the "true" output condition with positive output logic polarity. Operation is quite satisfactory up to about 100KHz. As may be seen this performance is realised with circuitry which is quite modest from the point of view of complexity and component cost; the two transistors employed are both low-cost general purpose types.

Shown in the photographs is a demonstrator panel which was built up be the author and which will now be briefly described. It mounts six gates wired to the circuit of figure 12, with their connection jacks and indicator lamps brought out conveniently to the front. Interested readers may care to build up a unit of this type in order to make themselves more familiar with practical logic circuit design and operation.

It may be noted that this panel with its six "NOR" gates is the first of a small number of digital demonstration panels which are introduced in the various theoretical chapters of this book. Although most of these panels will be individually useful in their own right, they have been designed so that they may be combined finally to produce a low cost digital demonstrator/experimental test-bed of some flexibility. The full demonstrator and its applications in tuitional and design work are described in chapters 12 and 13.

The six gates provided on this panel may be used to demonstrate quite a large number of logical principles, and to mock up many interesting logic functions. Figure 14 shows for illustration two configurations which may be used to demonstrate function synthesis. One configuration is that for the "exclusive-OR" function mentioned briefly in chapter 1; the other is that for the so-called "equivalence" function which plays an important part in digital arithmetic circuitry. Many other configurations may be found in the standard logic and computer texts, and in some of the books mentioned in the bibliography below.

Following chapters will show how the NOR gates described may also be used to demonstrate the basic operation of the "flip-flop" and the decoder. They are also used in the final demonstrator as binary readout units available for connection to the modules on the other panels.

SELECTED BIBLIOGRAPHY

CHRISTIANSON, D., "Integrated Circuits in Action, Part 5: In Search of the Ideal Logic Scheme," in **Electronics,** V.40, No. 5 (March 6, 1967), p. 149.

CLEARY, J. F., (Ed.) **G-E Transistor Manual,** 1964. The General Electric Co., Syracuse, NY. Chapters 5, 6, 7.

DEAN, K. J. **An Introduction to Counting Circuits and Transistor Circuit Logic,** 1964. Chapman and Hall Ltd., London.

HAWKER, P. D. T., "Transistor-Resistor Logical Circuits," in **Mullard Technical Communications, V.5,** No. 45, (Sept., 1960).

HENLY, H. R., "Economical Logic," in **Wireless World,** V.71, No. 10 (Oct., 1965) Also "Sequential Logic Design," in **W.W.,** V.73 No. 5 (May, 1967).

HILTON, A. M., **Logic, Computing Machines and Automation,** 1964. World Publishing Co., Cleveland (Meridian Paperback).

KENCH, E. J., (Ed.), **Integrated Logic Circuit Applications — Mullard FC Range,** 1967. Mullard Ltd., London.

POWERS, G., **"Diode-Transistor Micrologic."** Fairchild Semiconductor Applications Bulletin APP-107.
Also **"Counter Micrologic—A New Dimension in Multi-function Integrated Circuits."** Fairchild Semiconductor Applications Bulletin APP-118.

SAY, M. G., (Ed.) **Analogue and Digital Computers,** 1960. George Newnes Ltd., London.

SCOTT, N. R., **Analog and Digital Computer Technology,** 1960. McGraw Hill Book Co., Inc., New York.

WEBB, R. L., **"Components for Industrial Logic Circuits,"** reprint of paper presented at I.R.E.E. Convention, Canberra, 1965.

Chapter 3

COUNTING and NUMBER systems

Counting and scaling — two general types of counting circuitry — the gas-filled counter tube — the vacuum counter tube or "trochotron" — numerical notation systems the binary system — BCD codes — fixed and variable weighting — the reflected binary code — redundancy and error detection.

In a great many applications of digital techniques, circuitry is required to perform the operation of counting—usually, counting a series of electrical pulses. The pulses may be occurring randomly as in cases where they originate from a radiation detector such as a Geiger-Muller tube or where they correspond to physical articles on a conveyor belt interrupting a light beam; or, alternatively, they may be occurring periodically, as from an electronic or other oscillator.

The operation "counting" as used in electronics should be carefully differentiated from "addition," although quite often in digital equipment the circuitry used to perform these two operations may be somewhat similar.

Whereas **counting** is the operation of responding to a series of pulses and registering the number of these received, **addition** is the operation where the digital representations of two numbers are used to produce the representation of their arithmetical sum. Addition is one of a number of digital operations normally called "arithmetic," and as such is rather outside the scope of the present series of articles.

Another operation which should be differentiated from counting as such as "scaling." Here the connection between the two is much closer than between counting and addition, because in general exactly the same circuitry is used for scaling as for counting. The difference between scaling and counting is in terms of the interpretation placed on circuit operation.

In counting, the "output" of the circuitry is understood to be continuously available as a registration of the number of input pulses which have occurred up to the time concerned. In **scaling**, the circuitry is arranged to deliver an output pulse only after each successive occurrence of a given number of input pulses; it may be arranged to deliver an output pulse for every 5 input pulses, every 16 input pulses, or so on.

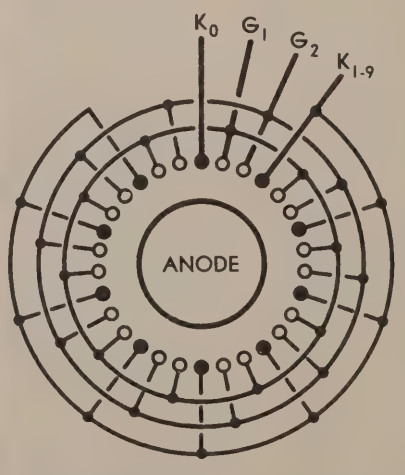
Where the input pulses are derived from an oscillator and are periodic in nature, scaling amounts to **frequency division** because the output of the scaler will also be periodic and will have a period corresponding to the designated number of input periods. Thus for periodic signals a scale-of-16 scaler becomes a $\times$ 16 frequency divider, etc.

The many applications of counting circuitry **as such** include digital-analog and analog-digital information conversion, sequential control of digital computer operations as directed by a 'program," and such digital instruments as voltmeters, frequency- and time period-meters, thermometers and frequency synthesisers. Some of these may be discussed in a later article of this series.

Counting circuits can be divided conveniently into two general types; those whose operation is based upon the use of circuit elements of a bi-stable nature, such as valve or transistor "flip-flops," and those whose operation is **not** based upon such elements. The latter type of counting circuitry is usually arranged to count directly in familiar decimal numeral notation, whereas special arrangements are often required if the former type is to count in this fashion. Usually it is arranged to perform the basic counting operation in binary notation, as will be discussed later in this article and in the following article.

Two examples of counting circuitry which do not depend upon bi-stable circuit elements will now be discussed. These are the circuits associated with the multi-cathode gas-filled counter tube or **"Dekatron"** (this word is actually a registered trade-mark of Ericsson Telephones Limited, but is commonly used to describe any tube of this general type) and those associated with the vacuum-type counter tube or **trochotron.**

Gas-filled counter tubes usually have a construction as shown in figure 1. A central disc anode is surrounded by some 30 rod electrodes spaced equidistantly around its periphery. Every third rod electrode is termed a "cathode," and all but one of these cathodes are connected together (K_{1-9}). The remaining cathode is brought out to a separate electrode (K_0).

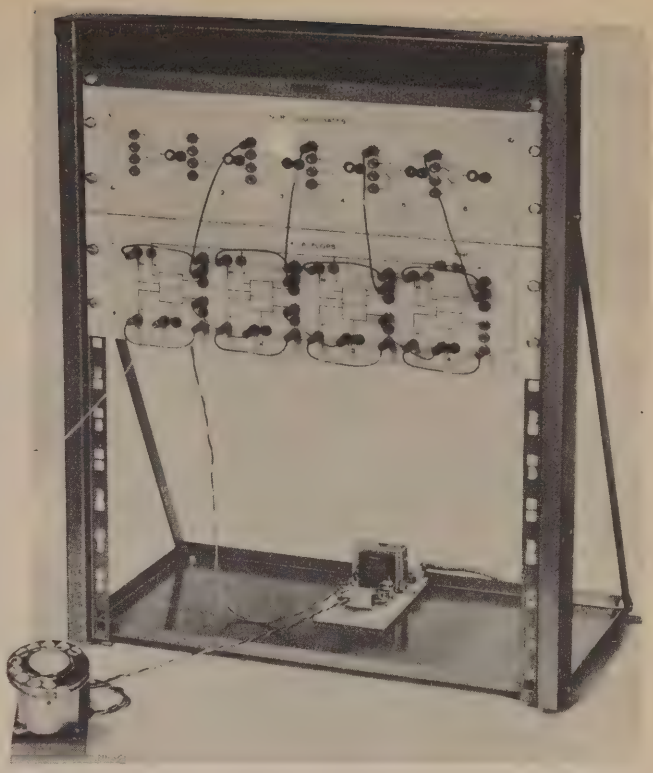


GAS-FILLED COUNTER TUBE

Fig. 1

The larger of the two "Dekatron" gas-filled tubes shown at left is a counter tube; the smaller is a selector tube. (Both courtesy Ducon Division, Plessey Components Group. "Dekatron" is a registered trade mark.)

In this chapter the author introduces the basic concepts of counting and number systems, and discusses decimal stepping tubes. The discussion is continued in chapter 4, which looks at counting techniques employing bi-stable circuit elements. Many of the concepts involved in counting are illustrated by the author's digital demonstrator unit, shown here partly complete. The unit is described in detail in chapters 12 and 13.



The rod electrodes immediately clockwise from each of the cathodes are all connected together, being called the "first guide electrodes" (G1). Similarly, the remaining electrodes are connected together to form the "second guide electrodes" (G2). The complete electrode assembly is mounted in a neon-filled glass bulb which is arranged so that any discharge between the anode and other electrodes may be observed visually.

The operation of this tube as a counter may be explained by reference to figure 2. The anode is connected to a positive supply voltage Vbb (usually about 400V) via a load resistor R1, while the cathodes are taken to earth—via a normally closed push button in the case of K_{1-9}, and either directly or via a small resistor in the case of K_0.

The guide electrodes G1 and G2 are connected to a source of positive voltage Vc (usually about 40V) to ensure that under steady-state conditions they do not act as cathodes. They are also inter-connected and AC coupled to the input terminal—G1 directly via a DC blocking capacitor only, and G2 via an integrating network formed by resistor R and capacitor C.

When voltage is applied to the circuit, pushing the reset button disconnects cathodes 1-9 from earth, leaving K_0 as the most negative of all the rod electrodes. A discharge thus occurs between this cathode and the anode, the current drawn lowering the anode voltage to a value determined by the **maintaining** voltage of the discharge. As this is lower than the **ignition** voltage of the remaining cathodes, the button may then be released without transferring the discharge away from K_0.

Consider now what happens when a **negative - going** pulse is applied to the input terminal, with an amplitude sufficient to temporarily cancel the positive bias on the G1 electrodes and drive them negative with respect to earth. When this occurs the A-G1 voltage will approach the ignition voltage; and as the G1 electrode nearest K_0 is in a region which is partially ionised, its ignition level will be least negative with respect to the anode. Thus, as soon as this voltage is reached the tube discharge will transfer from K_0 to this G1 electrode.

Now the input pulse lasts for only a short time, and when it ceases the G1 electrodes return to the positive potential Vc. If there were no G2 electrodes the discharge would simply return to K_0 when this occurred. However, there **are** G2 electrodes, and the integrating circuit formed by R and C has the effect of supplying these electrodes with a delayed version of the input pluse.

The integrating constants R and C are, in fact, so arranged that, at the very instant that the G1 electrodes return to the positive bias voltage, the G2 electrodes have just reached the voltage where the G2 electrode nearest the conducting G1 electrode is at its ignition voltage. Again, this G2 electrode will have the least negative ignition of all G2 electrodes because it is in a region of partial ionisation. Thus, the discharge will not return to K_0 but will transfer from G1 to G2.

When, in turn, the pulse on the G2 electrodes decays, the discharge cannot return to the G1 electrode from whence it came because this has returned to the positive voltage Vc. Instead, it transfers to the most negative electrode in the vicinity, which is also the electrode with the least negative ignition voltage—K_1. And it will remain on this cathode until another pulse appears at the input to initiate a second transfer cycle.

It may be seen that the single negative input pulse has resulted in a stable transfer of the tube discharge from one cathode to the next. In identical fashion, further input pulses will transfer the discharge around to the other cathodes in turn.

The tenth pulse will return the discharge to K_9; and if a resistor (shown

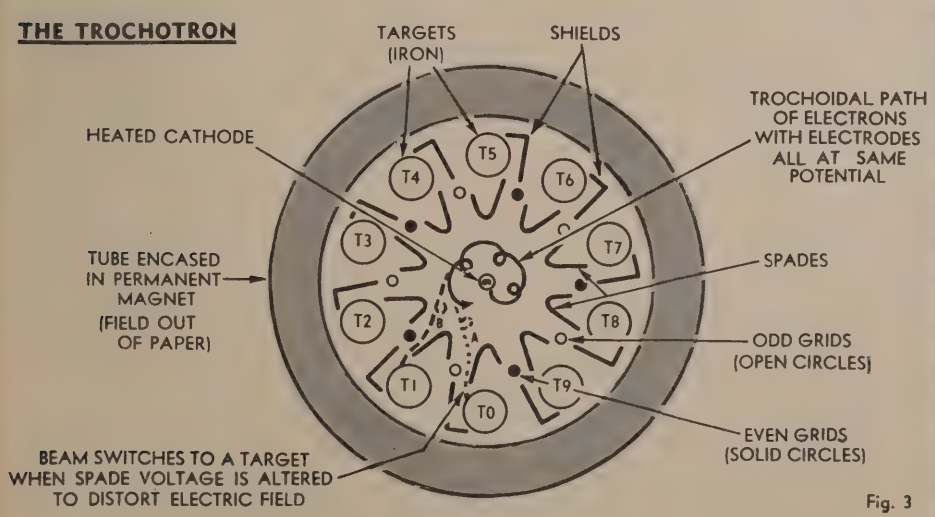


Fig. 3

The tube at right is a trochotron high-speed vacuum counting tube, capable of operating at counting speeds up to IMHz. (Courtesy Mullard-Australia Ltd.)

dashed) is fitted in series with this cathode a positive pulse wll be generated for possible use in feeding a "tens" counting circuit. As far as this stage is concerned, however, the discharge will continue to commutate from one cathode to another for each successive input pulse.

Read-out of the count registered by the tube may be performed at any time simply by looking into the tube to see which cathode is currently associated with the discharge—or, at least, this can be done when the tube is counting fairly slowly. When rapid counting is occurring it may not be possible to see which cathode is associated with the discharge glow.

And although the gas-filled counter tube is not capable of counting at particularly high speeds, the maximum speed is about 20KHz—rather too high for visual read-out while in operation! As with all counters, however, the read-out may be made whenever the input pulses slow down or cease temporarily.

The gas-filled counter tube may be used for scaling, as one might imagine. Even the tube shown in figures 1 and 2 may be used in this fashion, as the resistor from K_0 to earth provides an output for every 10 input pulses. However, for scaling purposes it is usual to employ a variation of the basic tube design in which other cathodes besides K_0 are brought out to separate connections. In some cases all 10 of the cathodes are given separate connections, in which case the tube is called a "selector" tube.

Finally, it may be noted that the gas-filled tube may be made to count in reverse simply by interchanging the connections to the G1 and G2 electrodes.

The major disadvantage of the gas-filled counter tube is that its counting speed is limited. Where counting must be performed at speeds higher than about 20KHz the de-ionisation time of the gas filling makes operation unreliable.

A vacuum-type counter tube which is capable of operating to about 2MHz is the **trochotron,** also called the "beam-switching tube" or the "beam-X switch." Although this tube is somewhat different from and more complex in its operation than the gas-filled counter tube, there are certain similarities which should become apparent from the following description.

Figure 3 shows the construction of a typical trochotron. In an evacuated envelope which is surrounded by an annular permanent magnet are ten cylindrical iron electrodes which act both as guides for the magnetic field and as "target" electrodes. The magnetic field within the tube may be visualised as parallel to the axis of the tube and emerging perpendicularly from the plane of the diagram. (In some versions of the trochotron there is no external magnet, but the target electrodes are themselves magnets.)

At the centre of the tube axis is a conventional heated cathode. Interposed between this cathode and each target electrode are ten "J"-shaped electrodes termed "spades," and adjacent to the spades wire electrodes termed "grids." Finally a set of "L"-shaped electrodes are included between the targets to act as electrical shields.

The shield electrodes are all connected together; while alternate grids are connected together to form two groups of five (shown as open and solid circles on the diagram, for clarity) termed the "odd" and "even" grids. All spades and targets are given individual connections.

In operation the shield electrodes are connected to a positive voltage supply (usually about 100V). The targets are connected to the same supply via low-value resistors, while the spades also connect to the supply via high-value resistors. The cathode is earthed, and the grid systems connected to a positive bias supply (about 30V) via suitable resistors. Figure 4 shows the general circuit configuration.

The tube is arranged so that with the correct operating voltages applied to the electrodes at switch-on, the magnetic and electric fields adjacent to the cathode are such that on leaving the cathode electrons do not move to any of the positive electrodes but spiral around the cathode in paths corresponding to the mathematical **trochoid** curve.

They do not proceed to the positive electrodes because the latter are all at the same voltage, resulting in a completely uniform radial electric field; the electrons are "trapped" into rotating around the cathode under the influence of the magnetic field, oscillating away from and toward the cathode under the combined influence of both fields.

To cause the tube to register "0" the electron beam must be guided into moving to Target T_0. This is done by applying a negative-going pulse to the "reset" terminal in figure 4, the pulse having an amplitude of approximately 35V. One of the effects of this is to apply the pulse to the "0" spade electrode, temporarily lowering the voltage on this electrode to earth potential or beyond.

With spade "0" at earth potential the electric field within the tube is distorted in such a way that the electron beam is deflected from its circular path to glance past spade 0 and be collected by target T_0. Although most of the electrons are collected by T_0, sufficient are collected by spade 0 to maintain the low voltage on this electrode due to voltage drop in the supply resistor. When the reset pulse ends, the beam thus remains in its new position, virtually "locking" itself to T_0 by maintaining the low voltage on spade 0.

To explain how the tube performs counting, reference must be made to the flip-flop unit shown capacitively coupled to the two grid systems and connected to the reset terminal.

The flip-flop will be discussed in some detail in the next article in this series and it is not proposed to explain its operation here. For the present, it will be sufficient to regard it as a device which responds to pulses applied to the input terminal by providing negative pulses alternatively to outputs A and B. The

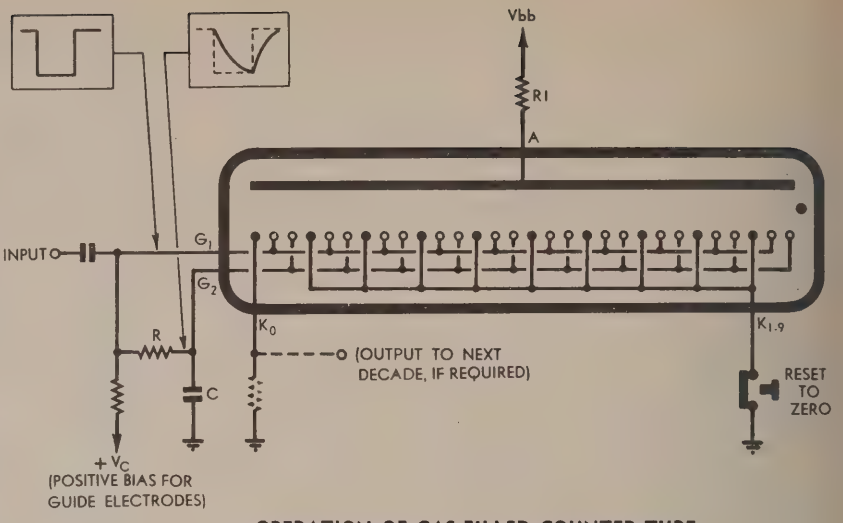


Fig. 2

OPERATION OF GAS-FILLED COUNTER TUBE

The basic circuit of a pulse-counting stage using a gas-filled counting tube of the type shown on page 44. By feeding each input pulse sequentially to the two guide electrode systems, the cathode discharge is switched.

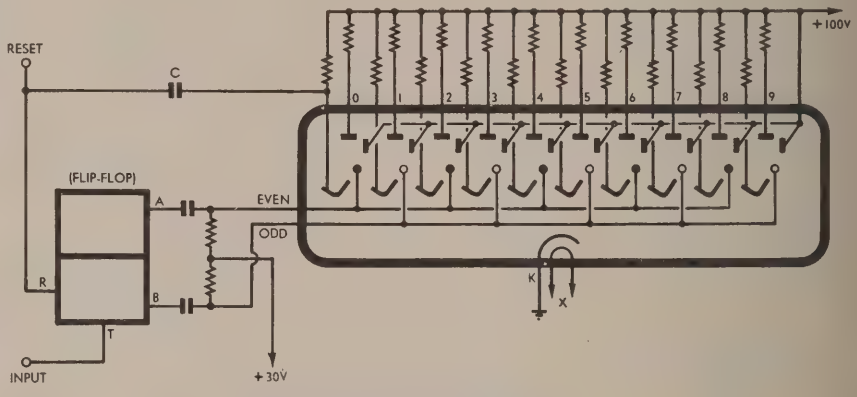


Fig. 4

OPERATION OF THE TROCHOTRON

The basic circuit of a counting stage using the trochotron vacuum counting tube. Input pulses initiate negative-going pulses alternatively at the "even" and "odd" grid electrodes, switching the electron beam.

application of the reset pulse to the flip-flop "R" terminal can be visualised as adjusting it so that the **first** input pulse will result in a negative pulse appearing at A.

When the first input pulse occurs, then, a negative pulse will be applied via the appropriate coupling capacitor to the trochotron "even" grids. As one of these is immediately adjacent to the electron beam currently switched to target T_0, this will cause the beam to be repelled from T_0; and the tube geometry and fields are so arranged that it is in fact deflected toward spade 1 and target T_1. As soon as some of the electrons are collected by spade 1, its voltage falls, and the beam is thence locked to target T_1 in the same manner as before.

When the second input pulse arrives it causes a negative pulse to be produced by the flip-flop, not at A but at B—and thence to the trochotron "odd" grids. And since one of **these** is adjacent to the beam in its new position, the beam will again be repelled and directed to spade 2 and target T_2. Fairly obviously, each input pulse will cause the beam to transfer to a new spade and target, while between pulses it will remain stably associated with the last spade and target to which it was switched.

And because the switching is being performed by a beam of electrons moving in a vacuum, the counting can occur at quite high speeds—up to about 2MHz, as mentioned before.

Note that there is no provision for direct visual read-out of the count; however, the targets are all brought out to separate connections to allow use to be made of the fact that there will be a voltage drop across the load resistor of the target currently associated with the electron beam. By suitable circuitry connected to the targets, the voltage drop at this target can be made to produce an indication.

Often a cold-cathode numerical indicator tube is used for this purpose; details of such indication devices will be given in a later chapter.

The fact that all 10 targets are brought out separately also makes the trochotron highly suitable for scaling. However, note that, due to the geometry of the tube and to the fixed magnetic field, it can switch only in one direction. It cannot be used for reverse counting and is, therefore, at a disadvantage compared with the gas-filled tube.

As mentioned earlier, the other main type of counting circuitry is that which is based upon bi-stable circuit elements. Nowadays this type of circuitry is actually in far more common use than the type which we have just been discussing; accordingly it will be discussed at somewhat greater length.

However, in order that the reader may be helped to a greater understanding than otherwise of the principles involved in such circuitry, it will be worthwhile at this point to re-examine our concepts of digital information and numerical notation.

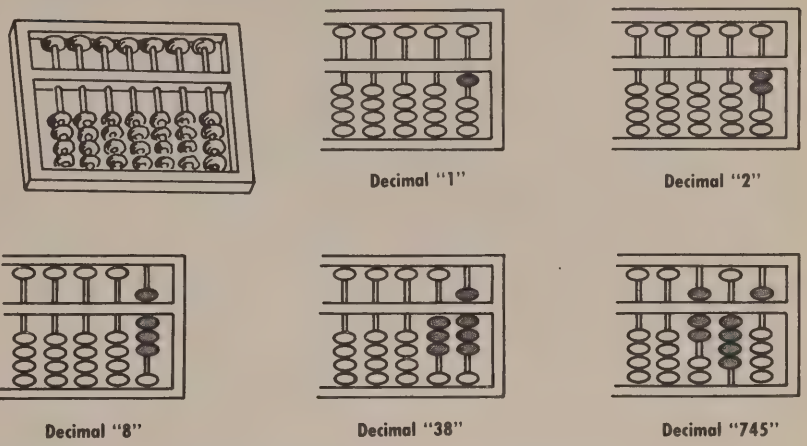
Thus far in the discussion of digital circuitry given in these articles we have regarded the digital information processed by the circuitry from a logical viewpoint—i.e., as **statements** which may take either of the two values "true" (1) or "false" (0). And this relatively simple conception of digital information is, in general, quite adequate for a consideration of the basic logical circuit functions.

However, it becomes necessary to analyse and expand this concept if one is to understand fully the operation of the digital circuitry involved in storage of information, counting, and "arithmetic." In the present chapter we will be concerned with the concepts involved in counting and information storage, while some of the concepts involved in "arithmetic" will be discussed in chapter 11.

In most cases the content of digital information statements is numerical: i.e., the information is in the form of numerical statements or numbers. And for a digital system having only two possible value-states to easily handle such statements, they should ideally be in the form of dyadic or "two-valued" numerical notation—**binary numbers.**

In order to be reasonably clear in one's conception of digital counting and storage it is, therefore, necessary to become familiar with the general concepts

The ABACUS: an early BCD counter



Although a very ancient invention, the "abacus" or bead counter is still very widely and effectively used for rapid addition and subtraction. A typical abacus employs "bi-quinary" notation, a system in which each decade is counted in two groups of five. The diagrams above should help in understanding how the abacus operates.

GLOSSARY OF IMPORTANT TERMS

Addition: A digital arithmetic operation in which the electrical representations of two (usually binary) numbers are used to produce the representation of their arithmetic sum. Circuit configurations involved in addition are discussed in chapter 11.

Base or Radix: That parameter of a numerical notation system which describes the number of values which may be taken in any given digital position. The base of the decimal system is 10, while that of the binary system used by most computers is 2.

BCD Codes: Numerical codes which employ the two binary digit values 1 and 0 in arbitrary combinations to represent the ten decimal digits. Theoretically there are more than seven million such binary-coded-decimal codes, but not all are used.

Bit: A convenient contraction of "binary digit." Having only two possible values (1 or 0), a single bit may be regarded as the smallest possible quantum of information.

Counting: A digital operation in which a configuration of circuit elements is arranged to produce an electrical representation of the number of pulses applied to their input. Counting plays an important part in the operation of most digital instruments.

Redundancy: An information theory parameter describing the efficiency of a symbolic information transfer system. Defined as the ratio of the unused information potential of the symbols being used to the information capacity actually being utilised.

Scaling: An operation in which a group of circuit elements are arranged to produce an output pulse after each successive occurrence of a given number of input pulses. When the input pulses are periodic in nature, scaling becomes **frequency division.**

of numerical notation, with the practical, notation systems other than the familiar decimal system and, in particular with the binary system.

The decimal numerical system to which most of us are so accustomed, with its 10 digits, is simply one of an infinite number of alternative arbitrary systems for conceiving and symbolising numerical quantities.

It is by no means the only system which has found practical use: the so-called "sterling" monetary system now passing out of use employs a combination of the **duodecimal** (based on groups of 12) and the **vigesimal** (base 20) systems, while our long-established time-measuring system divides the day into 12 hours (a duodecimal division) and the hours and minutes into 60 parts.

Similarly, in some countries counting was once performed in the **quinary** (base five) system from which originated the bead-counting "abacus."

Most of these systems use the so-called decimal digit symbols, i.e., the signs, 0, 1, 2, 3 . . . 9. Yet these are just as arbitrary as the scaling system adopted. The Romans, as is well known, used a different set of symbols: I, V, X, L, etc.; and the Arabs, from whom we inherited most of our present symbols, used a dot to symbolise zero instead of our 0.

In general, a practical numeral system adopts a quantity which becomes its base or "radix"; the base of the decimal system is 10, that of the quinary system is five, of the binary system two, and so on. Then it gives symbols to the integral quantities encompassed by the base quantity — i.e., "0"—"9", "0"—"4" and "0"—"1" respectively in the decimal, quinary and binary systems.

To cope with quantities larger than the base quantity and also with sub-integral quantities, the most efficient and practical course is then to employ the "principle of position"—whereby the absolute magnitude signified by any of the symbols depends upon its position with respect to a datum or reference mark. Usually the datum mark is a dot or point—hence the "decimal point," the "binary point" and so on—and the various positions which the symbols may occupy either side of this point form a power series using the same base as the system.

Thus, in the decimal system we may symbolise a certain quantity as "423.0," revealing by the spoken language equivalent, "four hundred and twenty-three," that the quantity signified is:

$$423.0=(4\times10^{2})+(2\times10^{1})+(3\times10^{0})+(0\times10^{-1}) \quad \ldots (1)$$

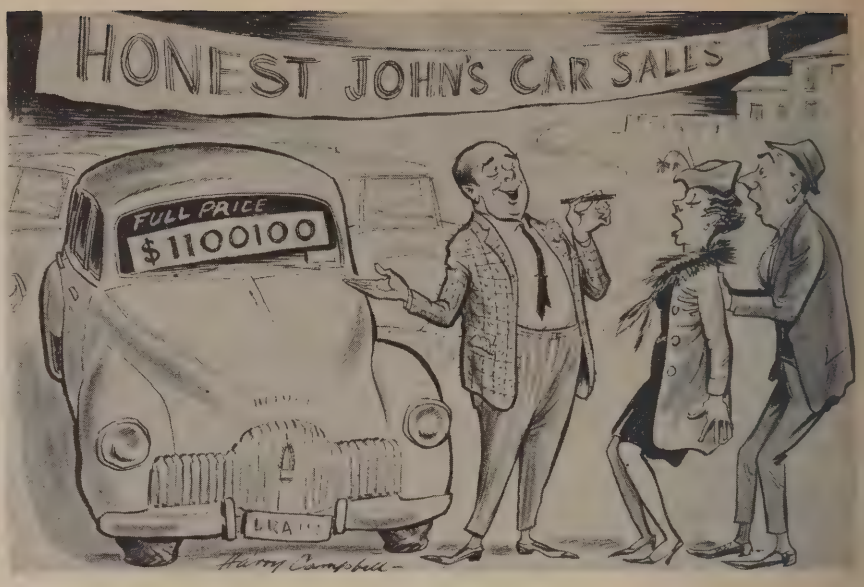
That is, in this system the positions to the left of the "decimal" point increase in positive orders or **powers** of the base 10, while those to the right of the point "increase" in negative powers.

In the quinary system the same set of symbols "423.0" means a quite different thing, although the same principles are in operation; the difference is that the positions either side of the "quinary" point here represent powers of five rather than of 10. Thus:

$$423.0=(4\times5^{2})+(2\times5^{1})\times(3\times5^{0})+(0\times5^{-1}) \quad \ldots (2)$$

Readers may care to check that the quantity thus symbolised in quinary notation is equivalent to "113" in decimal notation.

Because in the **binary** system there are only two integer symbols required to signify the concepts "zero" and "one",



". . . of course, that's binary notation."

(In numerical notation, it's all a matter of convention . . .)

in this system the set of signs "423.0" is meaningless. (Or at least half of it must be so: we can, if we wish, make any one of the symbols "4", "2", "3" or "0" signify "one", and signify "zero" by any other, but any way we did this would still leave two meaningless signs). It is usual to employ the familiar signs "0" and "1" to signify the concepts "zero" and "one" in binary notation, as is done in decimal notation.

To signify the quantity denoted by the decimal symbols "423.0" in binary notation it is simply necessary to adopt a parallel procedure to that in the decimal and quinary systems, in which the positions occupied by the symbols are ordered in a power series: in this case the series will have a base of two. The number will turn out to be "110100111.0"; i.e.,

$$\begin{aligned}110100111.0=&(1\times2^{8})+(1\times2^{7})+(0\times2^{6})\\&+(1\times2^{5})+(0\times2^{4})+\\&(0\times2^{3})+(1\times2^{2})+(1\times2^{1})\\&+(1\times2^{0})+(0\times2^{-1})\\=&\text{decimal } 423.0 \quad \ldots (3)\end{aligned}$$

It may be seen that although binary notation requires greater numbers of its symbols than does any other system (compare "423" with "110100111"), each position which the symbols may occupy has only two possible values. It is this characteristic which makes binary notation ideally suited to the representation of numbers processed by electronic circuits, as mentioned earlier, because circuitry designed to have only two possible value-states has maximum reliability and predictability of operation.

And while binary notation is less economic than others with regard to the number of symbols required to represent a given number, its economy of symbol variety leads to a quite radical simplification of arithmetic. For example, the laws of addition reduce to

$$\begin{aligned}&0+0=0\\&0+1=1+0=1\\&1+1=0 \text{ and carry } 1 \quad \ldots (4)\end{aligned}$$

Astute readers may note a resemblance between these laws and those of Boolean algebra of logic given in the first article of this series: a small demonstration of the Russell-Whitehead postulate that logic is the foundation of mathematics.

However, to continue: Although it is often most convenient to use binary notation to represent numbers within a digital system, this does not mean that the numbers need necessarily be fed into or out of the system in this notation. Although the latter can be done if required, it is usually quite easy to arrange decimal-binary and binary-

decimal translation logic so that the input and output to the system can be in familiar decimal form. This, despite the fact that the information is handled almost exclusively in binary form by the system in performing the required logical, counting or arithmetic operations.

It will be to our purpose to examine the binary "codes" used to represent decimal numbers in digital circuitry, as such codes are often involved in read-out of bi-stable counting circuits.

The binary notation which has been considered to this point is what is often termed "pure binary" notation — in which, as we have seen, the positions which the binary digits 0 and 1 may occupy represent magnitudes forming a power series to the base 2. Yet this is by no means the only way of ordering a binary scale using the same two digits; for some purposes it proves more convenient to employ a binary notation which uses a different ordering.

An example of a situation where it is often preferable to use other than pure binary notation is where the input and output to a system must be in decimal form, necessitating the decimal-binary and binary-decimal translation procedures to which mention has been made. In many cases this translation is easier if the binary notation used is other than "pure binary."

Irrespective of the binary notation employed, a minimum of four binary digits or **"bits"** are required to represent the 10 decimal digits; three binary bits are not sufficient, as these would provide only eight different bit combinations. As the following table of "pure binary" equivalents to decimals 0-15 illustrates, four binary bits provide 16 available different bit combinations.

Decimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

To represent the decimal numbers 0-9, only 10 of these 16 combinations are required; and from a general theoretical viewpoint **any** group of 10 of the combinations is equally capable of representing the decimal numbers. Thus theoretically there are quite a large number of available different binary-coded-decimal or "BCD" codes—some 7.6×10^7, in fact. Not all of these are used (!), but it will be worthwhile to look briefly here at a few of those more commonly used.

The "pure binary" equivalents to the decimal digits—i.e., those above the dashed line in the above table — are known as "8421 BCD." They are given this name to signify that the four binary bit positions are "worth" or have the **weight** of decimal digits 8, 4, 2 and 1 respectively. The main use for this code is for binary arithmetic.

A code which is often used in digital measuring instruments because of the relative ease in designing counting circuits which use it is "2421 BCD," where the weight of the binary bit positions are 2, 4, 2 and 1 respectively. It is identical with 8421 BCD for decimal digits 0-7, but uses the pure binary numbers 1110 (decimal 14) and 1111 (decimal 15) to represent decimals 8 and 9.

Where numbers must be stored in devices which must consume appreciable power it is convenient to employ a code using the minimum number of "ones" (as "1" is usually associated with a conductive state) to conserve power. A code which is useful from this point of view is "7421 BCD," in which no representation of a decimal number uses more than two 1's.

Again, in some types of arithmetical operation it is desirable that the code used exhibits a characteristic known as "9's complementation by bit complementation." By this is meant that the complement of a decimal number X with regard to decimal 9 (i.e., the number $9-X$) is formed simply by replacing all four of the binary bits representing X with their individual complements.

For example in a code which exhibits this characteristic the bits representing decimal 3 will be the complements of those which represent $(9-3)=6$, and so on. A code which has 9's complementation by bit complementation is the "4311 BCD" code shown in the following table for comparison with 8421 BCD, 2421 BCD and a 7421 BCD:

Decim.	8421 BCD	2421 BCD	7421 BCD	4311 BCD
0	0000	0000	0000	0000
1	0001	0001	0001	0001
2	0010	0010	0010	0011
3	0011	0011	0011	0100
4	0100	0100	0100	1000
5	0101	0101	0101	0111
6	0110	0110	0110	1011
7	0111	0111	1000	1100
8	1000	1110	1001	1110
9	1001	1111	1010	1111

It should be noted that there are in fact more than one of some of the above codes. This arises because some of the weighting systems allow alternative representation for certain of the decimal numbers; for example in 4311 BCD, decimal "5" could be represented equally validly by "0111", "1001" or "1010". **There are in fact 18 different 4311 BCD codes.**

Note also that in any BCD code the representation of decimal numbers larger than 10 is carried out by using four-bit groups to represent each decimal digit in the number. But while the resulting binary number will look the same as a pure binary number, it will in fact have a completely different meaning. Whereas in pure binary

$$11100100=(1\times2^7)+(1\times2^6)+(1\times2^5)+(1\times2^2)$$
$$=\text{decimal } 228 \quad \ldots (4)$$

the same digits in 2421 BCD would have the meaning

$$11100100=(1110)+(0100)$$
$$=\text{decimal } 84 \quad \ldots (5)$$

All the binary codes considered so far have been what are known as "fixed weighting" codes, because each of the four bits used consistently carries a certain weight, as designated by the code name. It is by no means necessary for a code to have fixed weighting, however.

An example of a variable-weighting code is the so-called "reflected binary" code, in which the representations of successive decimal digits differ by only one bit. Reflected binary code finds considerable use in applications where analog quantities such as continuously varying voltages, pressures, temperatures, shaft positions, etc., must be encoded digitally or "digitised."

When changing from one reading to another in such applications, reflected binary code is less liable to error than other codes because it does not depend upon a number of bits changing value in exact synchronism. Codes in which more than one bit has to change in value in changing a reading tend to produce spurious readings during the transition, as it is generally not possible to guarantee that all bits change value at exactly the same time.

Reflected binary code is shown below compared with pure binary. The reader may care to check for himself that none of the bits has fixed weights and that only one bit changes value between successive decimal digits. Note in contrast that in pure binary all four bits change value when progressing from decimal 7 to decimal 8.

Decimal	Binary	Reflected BCD
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101

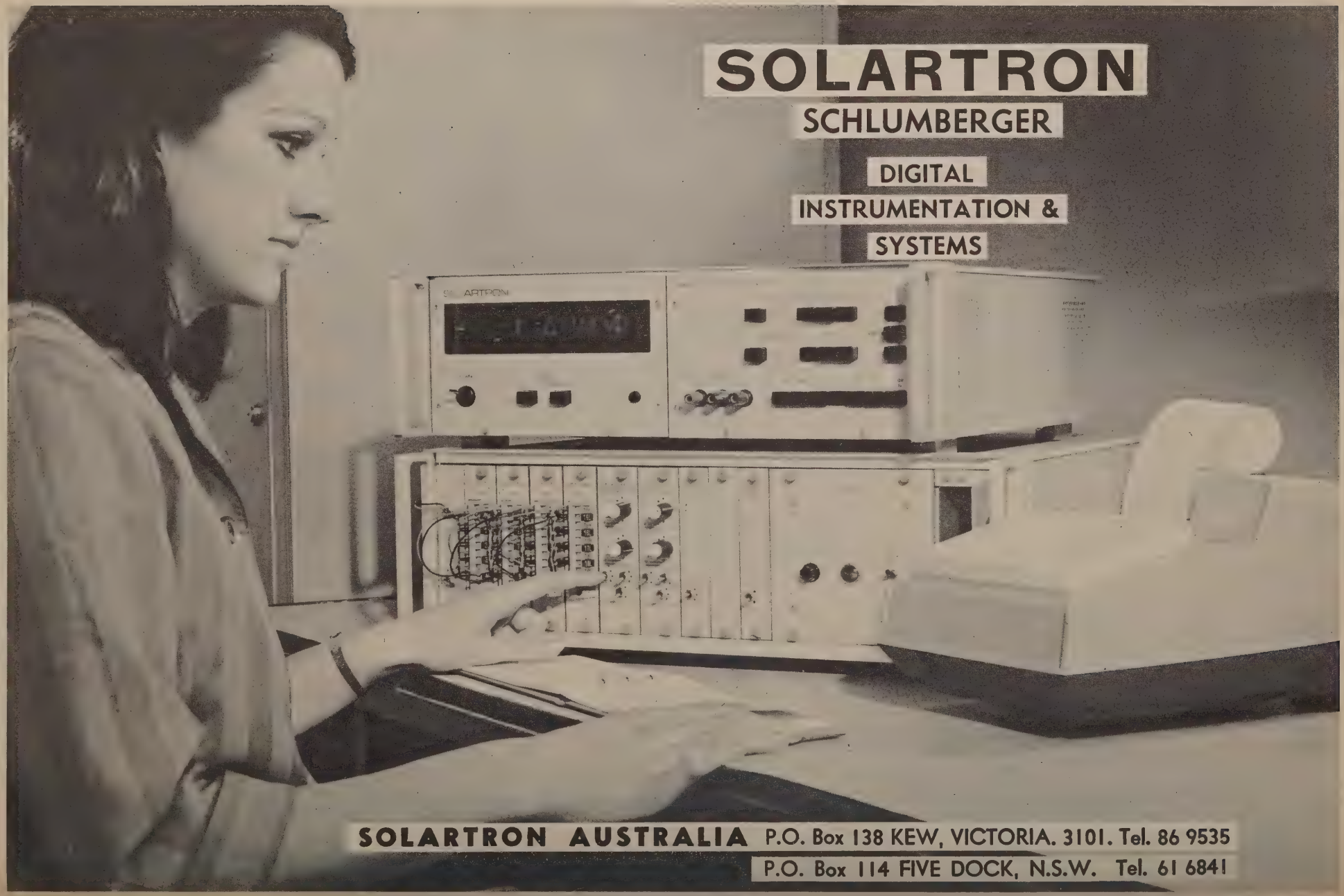
The reflected binary code is sometimes called the "Gray" code, in honour of its inventor Elisha Gray.

In concluding this necessarily rather brief discussion of numerical codes, mention should perhaps be made of methods

"SISSA'S REWARD": a problem in binary notation

In antiquity, according to legend, the Grand Vizier Sissa invented the game of chess for the Indian King, Shirham; invited by the King to name his reward, he asked only for the quantity of wheat required to cover a chess board by placing one grain on the first square, two on the second, four on the third, eight on the fourth, and so on. The King marvelled at Sissa's self-denial — until Sissa explained that the number of grains required would be only one less than the sixty-fourth power of 2: more than 18 million million, million . . .

2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7
2^8	2^9	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}
2^{16}	2^{17}	2^{18}	2^{19}	2^{20}	2^{21}	2^{22}	2^{23}
2^{24}	2^{25}	2^{26}	2^{27}	2^{28}	2^{29}	2^{30}	2^{31}
2^{32}	2^{33}	2^{34}	2^{35}	2^{36}	2^{37}	2^{38}	2^{39}
2^{40}	2^{41}	2^{42}	2^{43}	2^{44}	2^{45}	2^{46}	2^{47}
2^{48}	2^{49}	2^{50}	2^{51}	2^{52}	2^{53}	2^{54}	2^{55}
2^{56}	2^{57}	2^{58}	2^{59}	2^{60}	2^{61}	2^{62}	2^{63}


SOLARTRON
SCHLUMBERGER
DIGITAL
INSTRUMENTATION &
SYSTEMS
SOLARTRON AUSTRALIA P.O. Box 138 KEW, VICTORIA. 3101. Tel. 86 9535
P.O. Box 114 FIVE DOCK, N.S.W. Tel. 61 6841

used to detect errors which inevitably occur in coding due to switching inaccuracies, transmission noise, brief failures in the system, and so on.

In general, error detection and correction makes use of what is called information **redundancy**, where the redundancy of a piece of information or "message" is defined as the ratio of the unused information potential of its symbols to the capacity actually being utilised. Thus if a notation system employs a set of symbols giving "M" possible different symbol combinations, but only uses "N" of these symbol combinations to represent meaningful messages, it has a redundancy "r" given by the equation

$$r = \frac{\log_2 M - \log_2 N}{\log_2 M} \quad \ldots (6)$$

From this it may be seen that redundancy is virtually the opposite of symbolic economy. If we use all the possible combinations of a set of symbols, we obtain a code with maximum economy of symbols—using the least number of symbols to carry a certain message—but the code will conversely have zero redundancy. As every combination of the symbols will have a meaning, any error which changes the symbols of a message will produce a new message which is equally feasible as far as the receiver of the message is concerned. There will be no possibility that such an error may be detected by its production of a "silly" or meaningless combination.

An example of a code with zero redundancy is pure binary notation (but not 8421 BCD).

Increasing the redundancy of a code reduces the symbolic economy by making it necessary to use more than the absolute minimum number of symbols to carry a certain message. However, the higher the redundancy the higher the probability that errors may be detected because of the production of meaningless symbol combinations. And if the redundancy is increased above about 0.5, it often proves possible for errors to be not only detected but corrected as well.

An example of the latter is our own English language. This has a very high average redundancy, because while the number of "sensible" combinations of its 26 letters is large, it is still a small fraction of the total number of possible combinations. Thus, it is not particularly hard to decipher the following distorted message, despite its nine character errors in a total of 48:

"Im wintar thc lakis anf rivers ohtem freez ober."

The BCD codes discussed in this article all have a small but useful redundancy, because they use four binary bits to represent the ten decimal digits when, as we saw earlier, four bits are capable of representing a total of 16 numbers. In theory, a total of only 3.32 bits would be required to represent the decimal digits, so that using the equation given above, the four-bit BCD codes have a redundancy of:

$$r = \frac{4 - 3.32}{4} = 0.17 \quad \ldots (7)$$

In other words, whichever 10 of the 16 possible four-bit combinations are used in a BCD code, there remain six "forbidden" or meaningless combinations whose appearance indicates that an error has occurred. But fairly obviously there are many possible errors which could occur **without** producing one of these combinations; thus a redundancy of 0.17 as possessed by the four BCD codes, while useful, is by no means sufficient to detect all possible errors.

There are a number of techniques by which the redundancy is increased to allow error detection and correction. Some codes use an additional binary bit called the "parity" bit, and adopt a convention that the parity bit always takes a value to make the total number of "1's" in every group an even number (or always an odd number). This system will show up any errors which produce a single change in the total number of 1's in each group.

Another technique is to add additional bits whose value is arranged so that each group has a certain fixed number of 1's and 0s. Thus, there is the "2-out-of-5" code in which two, and only two, of the five bits in each group are 1's, and the "2-out-of-7" or **biquinary** code in which two and only two of each group of seven bits are 1's. A more detailed discussion of these and similar techniques is beyond the scope of this book, but it is hoped that the foregoing will provide useful insight into the principles involved.

SELECTED BIBLIOGRAPHY

DEAN, K. J. **Digital Instruments**, 1965. Chapman and Hall Ltd., London.

HILTON, A. M., **Logic, Computing Machines and Automation**, 1964. World Publishing Co., Cleveland, Ohio. ("Meridian" paperback).

JEYNES, G. F., "Decade Stepping Tubes and Their Operation," in **Mullard Technical Communications**, Vol. 4, No. 37, February 1959.

MILLMAN, J., and TAUB, H., **Pulse, Digital and Switching Waveforms**, 1965. McGraw-Hill Book Co., New York.

SAY, M. G., **Analogue and Digital Computers**, 1960. George Newnes Ltd., London.

Chapter 4

COUNTING with the "FLIP-FLOP"

A bi-stable element from two NOR gates—
the simple R-S flip-flop—the gated R-S flip-flop—
the J-K flip-flop — "ripple-carry" counting —
forward, reverse, and bi-directional counting.

It may be recalled that at the end of the preceding chapter we discussed numerical notation and encoding systems, following a brief examination of the type of counting circuitry which does not employ circuit elements of a bi-stable nature. Let us now turn our attention to the type of counting circuitry which nowadays is the more common: that which **does** employ bi-stable circuit elements. An appropriate place to commence our discussion is an examination of the bi-stable circuit elements themselves.

As the name suggests, "bi-stable" circuit elements are circuit sub-sections which have two (or alternative) stable operating states. External conditions may be used to switch the element from one of these states to the other, and back again as desired; but between changes in the external conditions the element retains the operating conditions which resulted from the last switch-over — in short, it has "memory" or **storage.**

Quite a few devices and circuit configurations have two stable states. The tunnel diode may be operated in this fashion, as may neon bulbs and PNPN diodes. Similarly, devices like toroidal ferrite cores may be arranged to be stably magnetised in either direction. But probably the most common bi-stable element is the valve or transistor **"flip-flop,"** also referred to as the "bi-stable," "multi" or "binary."

The flip-flop may be considered to be equivalent to two interconnected NOR gates. Figure 1 shows the idea; the output of each NOR gate is connected to one input of the other gate.

A moment's consideration of this diagram should show that only one NOR gate can be "on" at any particular time, where being "on" is taken to imply the presence of a logical "1" at the output. (If the positive logical polarity is adopted, 1=+9V, 0=0V, etc.). When gate A produces a 1 at M, this provides gate B with a 1 input, forcing it to produce a 0 at N; similarly, if B produces a 1 at N, this forces A to produce a 0 at M.

Whichever gate produces a 1 at its output forces the other to produce a 0; conversely, the gate which produces a 0 maintains the other in producing a 1. The arrangement is quite symmetrical, so that both possible states are stable.

To switch the circuit from one state to the other it is simply necessary to apply a 1 pulse to another input of whichever gate is currently producing a 1 output. Thus, if gate A happens to be on, application of a 1 pulse to lead X will force it to turn off—producing a 0 at M and thus allowing gate B to turn on. This will, in turn, produce a 1 at N, maintaining the new state of affairs after the external input pulse at X ceases. Similarly, the circuit may be switched back again by application of another 1 pulse to lead Y.

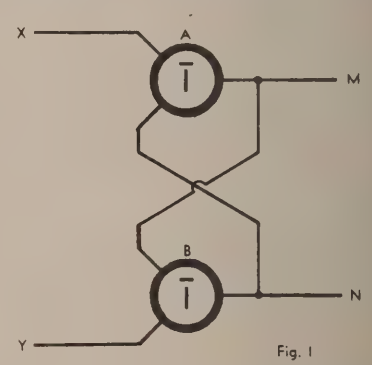


Fig. 1

Readers who constructed the NOR gate panel described in the second article can demonstrate the above operation for themselves quite easily by connecting two of their gates as in figure 1. As these gates use positive logic, application of +9V (=1) to a spare input of the "on" gate will be seen to produce switch-over.

The arrangement shown in figure 1 represents the simplest type of flip-flop element, often called the "reset-set" or **"R-S flip-flop."** This type of flip-flop is used mainly for storage of binary information; having two possible states, it can store one binary digit or **bit** by representing "zero" by one state or "one" by the other.

The circuit for a simple transistor R-S flip-flop is shown in figure 2, together with its usual logic symbol. As may be seen, the circuit is virtually that of two interconnected NOR gates of the resistor-transistor or "RTL" variety. The lead coding indicates that a logical 1 applied to a "set" (S) input tends to produce a logical 1 at output Y, whereas a 1 applied to a "reset" (R) input tends to produce a 0 at Y and a 1 at the complementary output.

Note that both DC level and pulse inputs may be provided, but that the logical convention used for pulse inputs is negative rather than positive as is used for both input and output DC levels. This is because with NPN transistors pulse input signals are made neg-

These modern digital instruments illustrate typical applications of the bi-stable counting elements introduced in this chapter. Chapter 10 discusses such instruments in some detail. Courtesy Hewlett-Packard Australia Pty. Ltd.)

ative-going. The reason for this is that with pulse input signals it is generally easier to initiate switch-over of the flip-flop by applying a "turn-off" pulse to the conducting transistor than by applying a "turn-on" pulse to the non-conducting transistor.

As may be seen, logically equivalent pulse and DC level inputs are applied to opposite transistor bases. Thus to "set" the flip-flop (i.e., to produce Y=1), the DC level S signal is a +9V step applied to the left-hand transistor; but a −9V pulse applied to the right-hand transistor via the alternative S input will produce the same effect.

If a circuit of the type shown in figure 2 is required to operate at high speeds, it is common practice to fit suitably selected small capacitors in parallel with the cross-coupling resistors as indicated. Such "speedup" or "commutating" capacitors maintain the flip-flop loop gain at high frequencies by compensating for the transistor input capacitance, and allow the circuit to switch over rapidly.

The R-S flip-flop is not particularly suitable for counting purposes because its "input" virtually alternates between the R and S leads every time the circuit switches over. In order to change the state one must know its existing state, because application of the input 1 (either DC level or pulse) to the inappropriate input will have no effect.

It is not feasible to apply the input to both leads simultaneously in the hope that the flip-flop will simply "ignore" the inappropriate signal, because this has a quite indeterminate outcome. A "race" condition is produced by interaction between the input signals and the internal switching action.

The above behaviour can be defined conveniently and economically by a "truth table" of the same type which we used to define the operation of logic circuitry. Thus if the subscript "n" is used to indicate the values of the flip-flop outputs prior to the application of the inputs R and/or S (either DC or pulse) and the subscript "n+1" to indicate values afterwards, the following truth table applies to the R-S flip-flop:

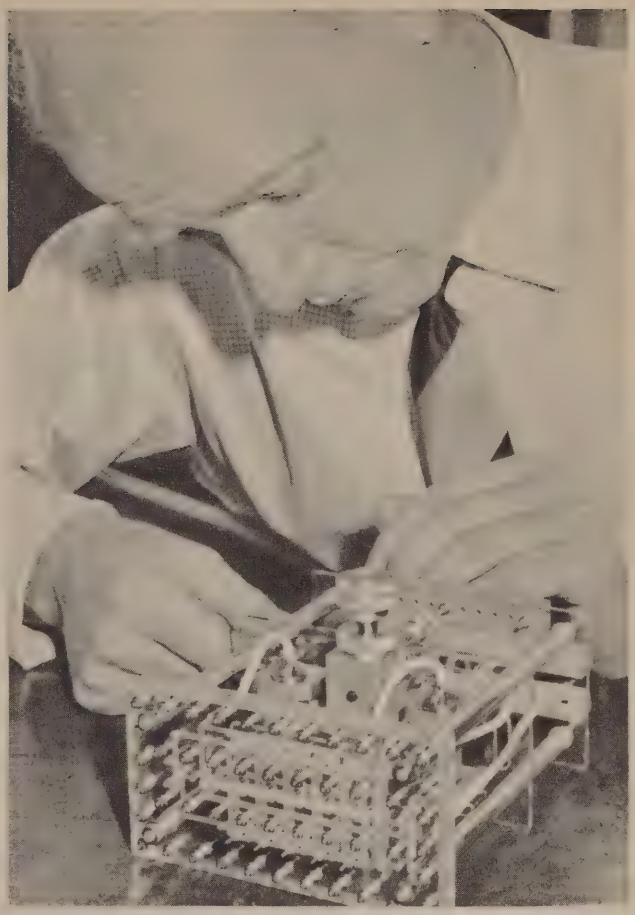
S	R	Y_n	$\bar{Y}_n$	Y_{n+1}	$\bar{Y}_{n+1}$
0	0	1	0	1 } Y_n	0 } $\bar{Y}_n$
0	0	0	1	0 }	1 }
1	0	1	0	1	0
1	0	0	1	1	0
0	1	1	0	0	1
0	1	0	1	0	1
1	1	1	0	?	?
1	1	0	1	?	?

For counting and many other purposes it is necessary to extend the basic R-S flip-flop by providing it with gating circuitry whereby the input (pulse) signals are guided to the appropriate input lead by control signals derived either from within the flip-flop itself or from external circuitry. A flip-flop provided with such gating circuitry is often called a "**gated** R-S flip-flop."

There are many ways of performing the input gating or "steering" function necessary to produce gated R-S flip-flop action. However the easiest to follow is probably that wherein the input pulses are guided to the appropriate transistor base by simple diode AND gates of the type shown in figure 3.

Here the input pulses, which are negative-going, are routed to each transistor base via a series capacitor and diode network. A resistor is connected to the junction of each capacitor and diode, and by applying DC gating signals to the resistors it is possible to "open" one of the gates and "close" the other — guiding the input pulse to the appropriate transistor. The logic polarity of the gating signals is **negative** (i.e., 1=0V, 0= +9V), because a positive control signal applied to a resistor will reverse-bias the diode concerned and thus "close" the gate.

Figure 4 shows how two such AND gates may be added to the basic circuit of figure 2 to produce a gated R-S flip-flop. The single pulse or "toggle"



Electronic circuitry is by no means the only medium available to the designer of logic and counting equipment. This new data processing system uses streams of compressed air. (Courtesy British Tele-communications Research Ltd.)

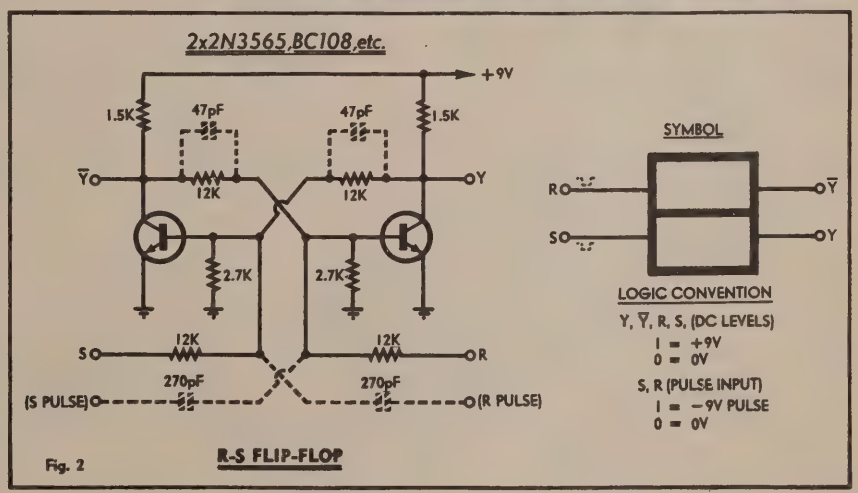


Fig. 2 **R-S FLIP-FLOP**

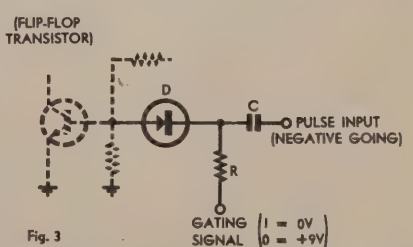


Fig. 3

input is labelled T, while the connections to the gating resistors are labelled "RG" (reset gate) and "SG" (set gate) respectively to indicate that in each case a control signal 1 (=0V) will cause the negative-going input pulse to be guided to the transistor base appropriate to produce either the reset (Y=0) or set (Y=1) state. The figure also shows the usual logic symbol for this type of flip-flop.

As the diagram suggests the gated R-S flip-flop may be provided with "direct" reset (DR) and set (DS) inputs in addition to the gated pulse input. These may be used both for "clearing" a counter of its registration and also for arranging for "filling" where a counter is arranged to count in a BCD code. Both these uses will be discussed in the following article.

As before, operation is speeded up by means of commutating capacitors across the cross-coupling resistors. With the circuit of figure 4 it also proves desirable to fit additional diodes across the gating resistor, as shown. The diodes allow the 270pF input capacitors to discharge quickly at the cessation of input pulses.

By virtue of its single pulse input and gating facilities, the gated R-S flip-flop has a great many uses in counting, logic, arithmetic and storage applications. However for some purposes it still has a shortcoming inherited from the simple R-S flip-flop. This is that the "indeterminate output" condition can still occur if ever the two gating signals are both at logical 1 (=0V) at the same time—allowing the input pulse to be fed to both transistor bases at once.

This may be seen conveniently by reference to the truth table definition of the gated R-S flip-flop shown below. Again the subscript "n" indicates values prior to the input pulse, and "n+1" indicates those afterward:

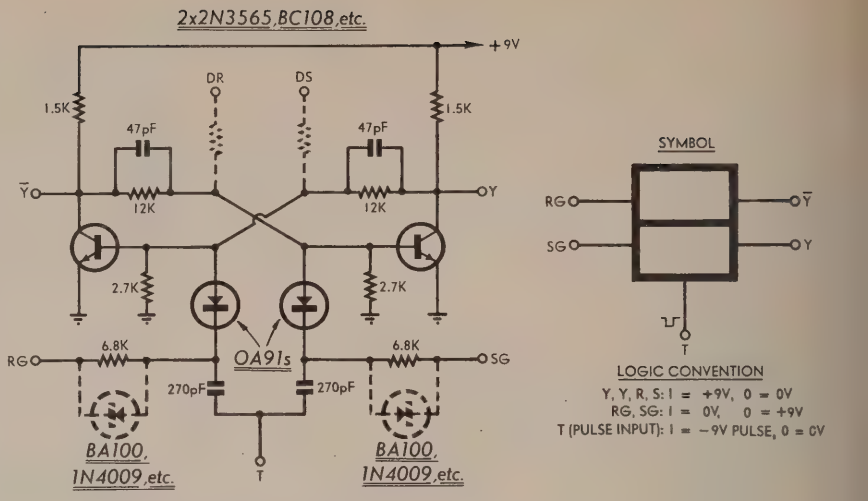
SG	RG	Y_n	$\bar{Y}_n$	Y_{n+1}	$\bar{Y}_{n+1}$
0	0	1	0	1 } Y_n	0 } $\bar{Y}_n$
0	0	0	1	0 }	1 }
1	0	1	0	1	0
1	0	0	1	1	0
0	1	1	0	0	1
0	1	0	1	0	1
1	1	1	0	?	?
1	1	0	1	?	?

The logic polarity conventions used here are indicated in figure 4.

In applications where the "ambiguous" RG=SG=1 situation can occur, the indeterminate behaviour of the gated R-S flip-flop in this situation can often be embarrassing, if not intolerable. For such applications it is therefore necessary to extend the basic flip-flop concept still further by equipping it with a means whereby it behaves in a predictable way when presented with such ambiguous "instructions."

A development from the gated R-S flip-flop which is provided with such a mechanism is the so-called **"J-K flip-flop."** This behaves exactly as the gated R-S flip-flop for all other conditions, but in the "ambiguous" situation it behaves predictably by "toggling" or "complementing"—i.e., switching over from its existing state to the other state.

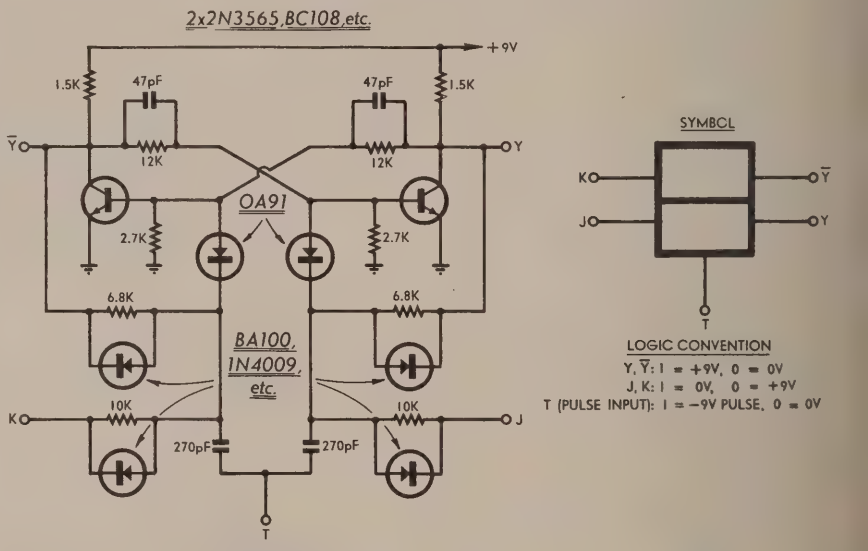
Figure 5 shows the circuit of a simple J-K flip-flop developed from those shown in the previous diagrams, together with its usual logical symbol. It may be seen



GATED R-S FLIP-FLOP

Fig. 4

Above and below are simple transistor circuits showing two common developments from the basic R-S flip-flop.



J-K FLIP-FLOP

Fig. 5

to be rather similar to the gated R-S flip-flop; the main difference lies in an additional set of gating resistors and diodes, deriving internal gating signals from the J-K flip-flop's own output and complementary output. A small additional difference is that the external gating terminals are now labelled "J" and "K" rather than "SG" and "RG."

The function of the additional gating components may be visualised fairly easily: when J and K are both at logical 1 (=0V), the flip-flop provides its own gating. The output terminal at logical 1 (here equal to +9V, as the **output** logical convention is positive) closes the gate to its associated transistor, routing the input pulse to the other transistor.

As the latter will be the "on" transistor, the negative-going input pulse will thus turn it off and cause the flip-flop to switch over. Thus when J and K are at logical 1 the J-K flip-flop simply switches back and forth for each input pulse. This is termed the **toggling or complementing mode** of operation.

For the three other possible combinations of values for J and K, the circuit works in exactly the same way as did the gated R-S flip-flop. If both J and K are at logical 0 (=+9V), both input gates are closed and the flip-flop is unaffected by the input pulse; if one or the other is taken to logical 1, it opens the associated gate.

The truth-table definition for the J-K flip-flop thus becomes as below, where the subscripts "n" and "n+1" have the same meanings as before:

J	K	Y_n	$\bar{Y}_n$	Y_{n+1}	$\bar{Y}_{n+1}$
0	0	1	0	1 } Y_n	0 } $\bar{Y}_n$
0	0	0	1	0 }	1 }
1	0	1	0	1	0
1	0	0	1	1	0
0	1	1	0	0	1
0	1	0	1	0	1
1	1	1	0	0 } $\bar{Y}_n$	1 } Y_n
1	1	0	1	1 }	0 }

The logic polarity conventions used here are also the same as before. Note that the J-K flip-flop has a predictable output for all possible input contingencies; also that where the J and K inputs are not identical, the value of the Y output becomes identical with that of the J input terminal.

At this point it may be worthwhile to point out a fact which astute readers will no doubt have realised already. This is

that the gated R-S flip-flop can be arranged to perform the above self-gating function simply by connecting its SG and RG inputs to the Y and Y-complement outputs respectively. However, when this is done the gated R-S flip-flop becomes capable of **only** toggling or complementing mode operation, as it has only one set of gating components.

Thus it may be seen that while the gated R-S and J-K flip-flops may be equally suitable for a great many applications, the J-K variety has a greater number of potential applications by virtue of its greater flexibility and fully predictable behaviour. However, for most of the applications to be discussed in the remainder of this article the two types would be equally suitable.

In concluding this introductory discussion of flip-flops it should perhaps be noted that the simple circuits given in figures 2, 4 and 5 should by no means be taken to represent the only possible circuit configurations capable of performing the R-S, gated R-S or J-K functions. While quite practical and suitable for experimental and demonstrational purposes as will be discussed later, in the following article, they are in fact very simple examples of the three most common flip-flop types.

Commercial flip-flop elements are often considerably more sophisticated than these simple circuits, although the basic operation may be very similar. Partly the sophistication and complexity arises from the need to make elements which are capable of operating at extremely high speeds; however, circuit complexity sometimes proves a necessity in order to obviate the need for certain components which may be hard to provide within the selected method of fabrication.

An example of the latter occurs in the case of integrated microcircuit flip-flops of "monolithic" construction, where all circuit functions are performed by structures fabricated from a single chip of silicon typically measuring only 0.15in x 0.075in.

Because it proves difficult to provide even modest values of capacitance without increasing the size of the monolithic chip to an embarrassing size, the designer finds it necessary to alter the basic circuit configuration so that operation is achieved using only very small capacitance values — often the internal capacitance of transistor or diode elements.

Usually this involves extra transistor and diode functions; commercial microcircuit J-K flip-flops can thus involve as many as 30 micro-transistor elements, although some designs have less than half this figure. Commercial discrete-component and microcircuit flip-flops are pictured on this page.

Although the R-S, gated R-S and J-K flip-flops are probably the most common types encountered, they are by no means the only types. Apart from other basic types differing slightly from these three in mode of operation, there are also combination elements having virtually two or more basic flip-flops interconnected. However it will be found that familiarity with the R-S, gated R-S and J-K types will make it fairly easy to follow the operation of any other types encountered.

Having looked in some detail at the flip-flop as a representative bi-stable circuit element, then, let us now turn to examine some of the ways in which such bi-stable elements are used to perform counting. In general, there are two main methods of counting with bi-stable elements: so-called "ripple-carry" counting, and "shift" counting. The first of these will now be discussed.

Ripple-carry counting takes its name from a "rippling" or successive carry-over effect which occurs in this type of counter when the last of an even number of pulses enters the chain or **"register"** of bi-stable elements.

A ripple-carry counting register consists of a chain of bi-stable elements, each operating in the "toggling" or complementing mode and connected to the succeeding elements in a manner such that a pulse is passed on to the succeeding elements on every second switch-over. Figure 6 shows the logic diagram of a ripple-carry counting register using flip-flops.

As the flip-flops must operate in complementing mode they may be either J-K elements having the J and K inputs tied to logical 1, or gated R-S elements having the SG and RG inputs tied to the Y and Y-complement outputs respectively. Gated R-S elements are shown in the diagram of figure 6.

The operation of such a register is not hard to follow if it is remembered that the "T" input of a flip-flop is activated only by negative-going pulses. (It is quite feasible to design flip-flops to accept positive-going pulses; such flip-flops would simply involve polarity conventions opposite to that assumed in the following explanation.)

Prior to the arrival of input pulses, all elements are reset to Y=0 by feeding a logical 1 (=+9V) to their direct reset (DR) terminals from the common reset line via isolating diodes and resistors. The arrival of the first negative-going input pulse thus causes FF1 to complement, so that its Y output switches from logical 0 (=0V) to logical 1 (=+9V).

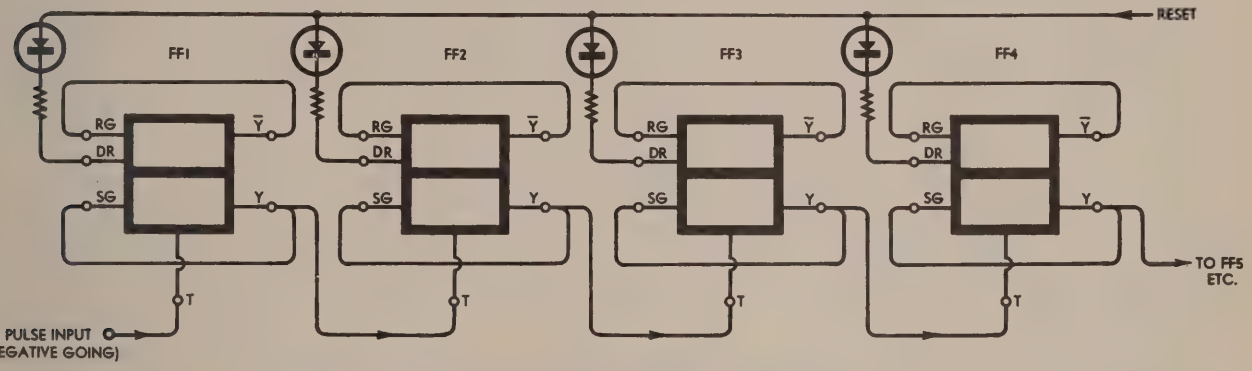
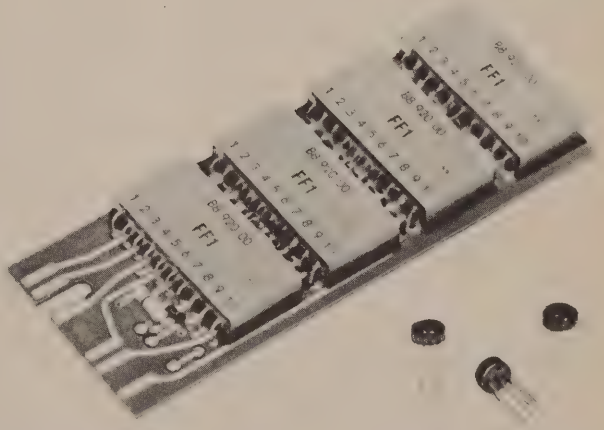


Fig. 6 **RIPPLE-CARRY "UP" COUNTER USING GATED R-S FLIP-FLOPS**

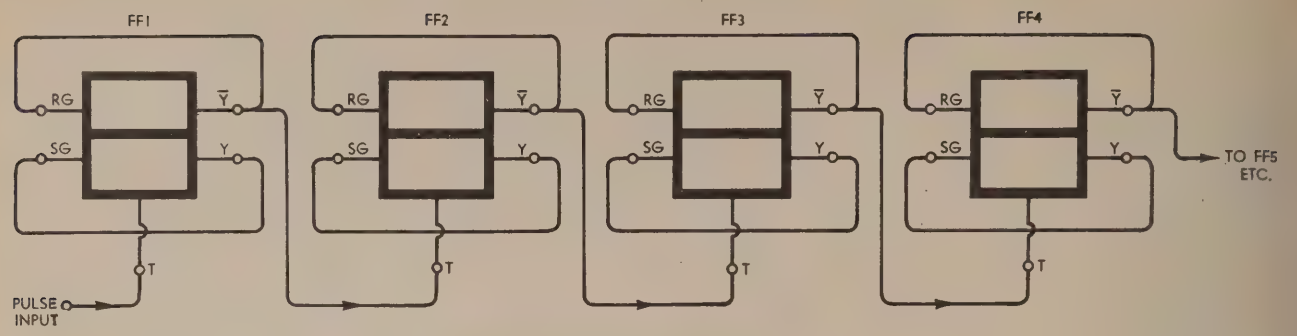
Repeated from an earlier article in the series, these pictures show typical commercial counting elements. The units mounted on the board are discrete - component flip-flops; the smaller units are integrated - microcircuit flip-flops. (Courtesy Mullard-Australia Pty. Ltd., and Fairchild Australia Pty. Ltd.)



Despite the interconnection between Y of FF1 and T of FF2 this transition does not affect FF2, because it is positive-going. Thus a single input pulse simply sets FF1 at Y=1 and leaves the remaining flip-flops at Y=0.

However, the next input pulse to arrive has a slightly more profound effect. It causes FF1 to complement again, returning its Y output to logical 0—and because **this** transition is negative-going, it provides a negative-going input pulse for FF2. Thus FF2 complements, its Y output switching from 0 to 1. However, as this is a positive-going transition no carry is propagated to FF3.

Thus after two input pulses have been received FF2 is in the "set" or Y=1 state, while all other flip-flops—including FF1—are in the "reset" or Y=0 state. (Note that the words "set" and "reset" are used to describe both the Y=1 and Y=0 states, as well as the



RIPPLE-CARRY "DOWN" COUNTER Fig. 7

When connected in this fashion, gated R-S or J-K elements perform pure-binary "down" counting.

operations involved in adjusting the flip-flop to these states.)

The third pulse to arrive produces as few transitions as did the first. It simply causes FF1 to set to Y=1 once more, leaving the register with FF1 and FF2 set but all remaining flip-flops in the reset state.

However, the fourth pulse to arrive has an even more profound effect than the second. First of all FF1 is caused to reset; but as this is a negative-going transition at the Y terminal, a pulse is propagated to FF2. As FF2 is already set (by the second input pulse), it therefore resets also; and as this in turn involves a negative-going transition of its Y terminal, a negative-going pulse is propagated to FF3. As the latter has to this time remained reset, it is thus complemented to the set position.

Thus after the fourth pulse the register is left with FF3 set but with all other elements including FF1 and FF2 reset.

In similar fashion the fifth input pulse leaves FF1 and FF3 set; the sixth pulse leaves FF2 and FF3 set; the seventh pulse leaves FF1, FF2 and FF3 all set; the eighth pulse resets all three to set FF4, and so on.

A little thought will show that every **odd** input pulse will complement FF1 alone, while every **even** pulse will complement (in turn) both FF1 and FF2. Similarly every **alternate** even pulse will complement FF1, FF2 and FF3 in turn, and every **fourth** even pulse will complement all four in turn. This can be conveniently represented by the table below, where the values in each flip-flop column represent the value of the Y output of that flip-flop following the input pulse concerned:

INPUT PULSE	FF4	FF3	FF2	FF1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0

The flip-flop positions are laterally transposed in the table to emphasise a fact which the astute reader may by this point have realised: that the simple ripple-carry bi-stable register counts in "pure binary" notation. Examining the table it will be seen that FF1 has the weight of a "1" register, FF2 that of a "2" register, FF3 that of a "4" register, FF4 that of an "8" register, and so on. As the four flip-flops represent four binary bits, they are capable of counting from decimal 0—15.

Thus in general a register of N flip-flops connected in this fashion will count up to M pulses, and the Nth element will have a binary weighting of W; where

$$M = 2^N - 1 \quad \ldots (1)$$

$$W = 2^{N-1} \quad \ldots (2)$$

After M pulses have been received, all N flip-flops will reset and the cycle will begin over again. From this it may be seen that the ripple-carry counter is quite suitable for **scaling** and **frequency division** as well as counting. When used for frequency division, each flip-flop virtually acts as a $\times 2$ frequency divider.

A ripple-carry counter of the type shown in figure 6 is known as an "up" or "feedforwards" counter because it gives a binary registration of the input pulses which **increases** in standard binary notation. However, by making a slight change in the flip-flop interconnections it is possible to produce a configuration which performs "down" or "reverse" counting—wherein the binary registration **decreases** with each successive input pulse.

Figure 7 shows the altered configuration. All that has been done is to take the carry-over signals in each case from the Y-complement terminals rather than

GLOSSARY OF IMPORTANT TERMS

Bi-stable: Any circuit element or configuration having the capacity to adopt one of two alternative stable states. In digital equipment bi-stable elements and configurations are used widely for information storage; each bi-stable can store one bit.

Flip-flops: Probably the most common electronic bi-stable element, usually comprising two or more transistors in a symmetrical circuit such that when one is conducting the other is virtually cut off. There are many different flip-flop variants, common types being the "R-S," "gated R-S" and "J-K" flip-flops.

Register: A group or configuration of bi-stable elements capable of storing a group of bits. Usually the bits stored are regarded as constituting a binary or binary-coded-decimal number.

Ripple-Carry Counting: A counting configuration in which each bi-stable counting element receives its input from the element next lower in numerical significance. Named because of the "rippling" wave of inter-element carry-overs which occurs upon the arrival of the last of an even number of input pulses.

Storage: Circuit elements capable of adopting one of a number of alternative stable states may be used to effectively store information for subsequent "retrieval." Digital storage is most often performed by bi-stable elements or configurations capable of storing information in binary form.

Toggling Mode: A mode of operation in which a bi-stable element is arranged to switch from each stable state to the other alternately upon the arrival of a series of pulses.

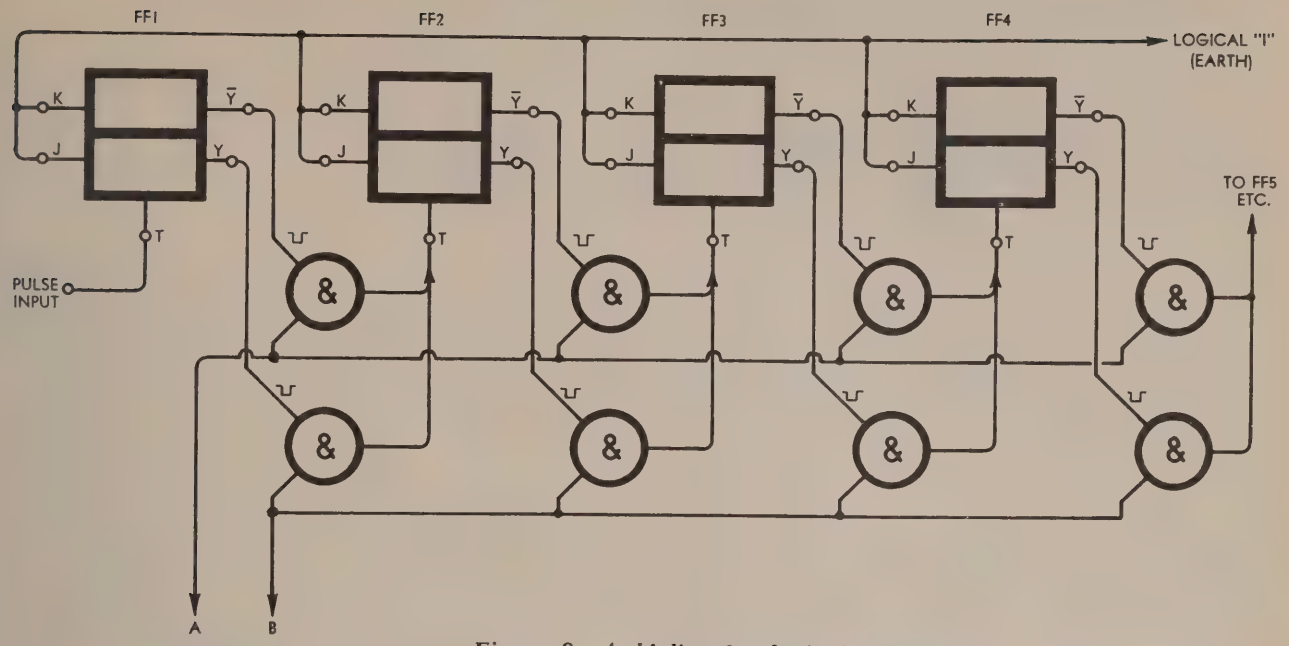


Figure 8: A bi-directional ripple-carry counting register using J-K flip-flops. By opening the upper set of AND gates, the register is made to perform "down" counting; alternatively by opening the lower set of gates it will perform "up" counting. The J-K elements are operated in complementing mode by connecting both J and K inputs to logical 1.

from the Y terminals. This produces carry-over on reset-set transitions rather than on set-reset transitions as occurred before. If readers care to check the effect of this they will find that each pulse will result in the **subtraction** of one bit from the binary registration.

Thus if the four-bit binary register shown is initially reset to 0000 (decimal 0, also decimal 16 and multiples thereof), the first input pulse will set all flip-flops to give 1111 (decimal 15). The second will reset FF1 to give 1110 (decimal 14), while the third will give 1101 (decimal 13); and so on.

It is possible to make a **bi-directional** ripple-carry counter by using two sets of AND gates to alternatively perform the carry-over interconnections. The logic diagram for such a counter is shown in figure 8.

Here for "up" or forward counting, a 1 is applied to line B and a 0 to line A, rendering the AND gates connected to the Y outputs operative. Alternatively for "down" or reverse counting a 1 is applied to the A line and a 0 to the B line—rendering the Y-complement gates operative. The AND gates used for this purpose might possibly be of the diode-capacitor type shown in figure 3.

Note that the flip-flops shown in figure 8 are of the J-K variety, with inputs J and K taken to logical 1 to produce the complementing mode of operation. This has been done both in the interests of clarity and also to illustrate how the J-K element is used for ripple-carry counting.

Readout of the count contained by a ripple-carry register is simply a matter of sensing the state of each of the bi-stable elements. This may be done by connecting low-loading indicating lamp circuitry to each flip-flop output, either directly for binary readout or via decoding circuitry for decimal readout.

Thus if the flip-flops used are those shown in figures 4 or 5 and described in the next article, binary readout may be performed using the indicating NOR gates provided on the first demonstrator panel. As the NOR elements contain inversion, they are connected to the Y-complement flip-flop outputs to indicate the desired Y=1 state.

Decoding circuitry and more elaborate readout methods will be discussed and described in later articles.

It should be apparent from the foregoing brief discussion of ripple-carry that this method is a relatively straightforward way of utilising bi-stable elements for binary counting. And as such, ripple-carry counting finds extensive use in digital instruments and computing.

However, the time delay involved in a long ripple-carry register, when the input pulse virtually "ripples" along the chain of elements, and the time necessary for all elements to settle down after the arrival of the last ofa large even number of pulses, make this method rather unsuitable for applications where counting must take place at extremely high speeds.

Although it is possible to reduce the delay of this type of counting circuitry by special techniques, it is usually simpler where high-speed operation is required to forsake the ripple-carry counting method in favour of the second main method, which as mentioned earlier is known as **shift counting.**

Shift counting techniques and the methods employed to perform binary-coded-decimal or "BCD" counting are discussed in the next chapter.

SELECTED BIBLIOGRAPHY

CLEARY, J. F. (Ed.), **G-E Transistor Manual, 1964.** Ch.s 5, 6, 7. The General Electric Company, Syracuse, New York.

DANCE, J. B., **Electronic Counting Circuits,** 1967. Iliffe Books Ltd., London.

DEAN, K. J., **An Introduction to Counting Circuits and Transistor Circuit Logic,** 1964. Chapman and Hall Ltd., London.

MILLMAN, J., and TAUB, H., **Pulse, Digital and Switching Waveforms,** 1965. McGraw-Hill Book Co., New York.

POWERS, G., **Diode-Transistor Micrologic.** Fairchild Semiconductor Applications Bulletin APP-107. Also **Micrologic Shift Counters.**

Chapter 5

SHIFT COUNTING, and BCD

Counting with bistable elements, continued: Shift counting — the shift register — the normal ring counter — the twisted ring counter — bistable counting and BCD codes — construction of a panel with four gated R-S or J-K flip-flops.

In the last chapter, you may recall, we introduced bi-stable circuit elements and looked at the three main types of flip-flop elements—the R-S, gated R-S and J-K varieties. We then discussed one of the two main ways in which such bi-stable elements may be used for counting, the so-called "ripple-carry" counting system. Let us now pass on to a consideration of the second system of counting using bi-stable elements: **"shift"** counting.

The principle of shift counting is that a series of bi-stable elements are interconnected in such a fashion that, when pulsed **simultaneously** by the input signal, they each behave in a fashion determined not by their own previous state, but by that of the preceding element. This may be used to shift a pattern of logical values along the register, so that the location of the pattern at any time may be used as a registration of the number of input pulses received.

Figure 1 illustrates the general principle involved, showing a section of a

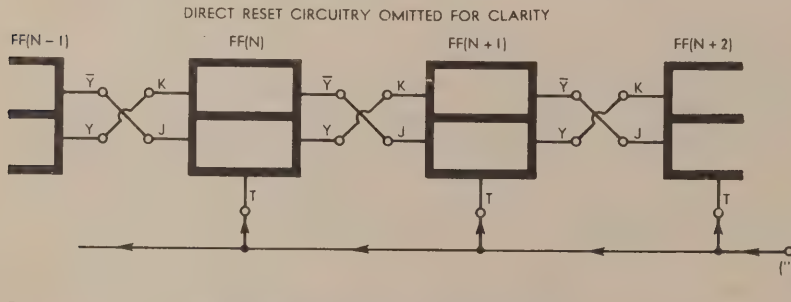


Fig. 1 **SHIFT REGISTER (USING J-K ELEMENTS)**

chain of elements connected in this fashion. J-K elements are shown, although gated R-S elements are in many cases equally suitable. Note that the input pulses (here often called "clock" pulses as they synchronise the transitions of all elements) are applied to all T terminals together.

The cross-connection of inputs and outputs shown (Y to K and Y-complement to J) determines that when activiated by the clock pulses, each element switches to the state previously occupied by the preceding element. Thus if element FF(N−1) is set to Y=1, the first clock pulse will force FF(N) to be similarly set, the second pulse will force FF(N + 1) to be set, and so on. Thus each clock pulse will cause each element to adopt the value previously held by the element at its left, and a series of pulses will progressively shift any pattern of values along the register from left to right.

It may be wondered why in this case the clock pulses are fed to the elements from the right rather than from the left; particularly so as the logical values are being shifted from left to right. This is done because it is impossible to prevent the self-inductance and stray capacitance of the clock pulse line from producing a slight delay or "clock skew" in the pulse as it passes along the line.

If the pulses were fed in from the left as before, the delay would result in each element being switched slightly after its predecessor. This would upset operation at high speeds, because for proper shifting action it is essential for an element to be able to sense the value held by its predecessor; this is hardly possible if the latter has already switched to its new value!

However, by feeding the pulses in from the right, clock skew simply causes each element to switch slightly **before** its predecessor—an effect which in no way upsets operation.

The cross-connections shown in figure 1, it may be noted, are electrical rather than logical: logically, the elements must be connected directly if each element is to adopt the value held by its predecessor. However, it may be remembered that with flip-flops of the type shown last month in figures 4 and 5, the logic polarity at the inputs is opposite to that of the outputs—hence for direct **logical** connections the **electrical** connections must be crossed as shown.

Any system of bi-stable elements connected in the manner shown in figure 1 forms what is commonly termed a "shift register." Quite apart from their uses in counting, such registers are widely used for storage, arithmetic and conversion operations on binary numbers.

It should be fairly evident that it would be a relatively straightforward matter to "shift" a long binary number into such a register, store it conveniently therein either while operating upon it or otherwise, and then "shift" it out the other end when desired. The number of elements in the register quite naturally determine the number of bits in the largest binary number that could be handled in this fashion.

As with the ripple-carry configuration the shift register can be made to operate in the reverse direction: in this case by deriving the input gating signals from the "succeeding" elements rather than the "preceding" ones. And similarly a bi-directional shift register can be produced by employing two sets of gates to control which of the two sets of gating signals control the elements at any desired time. Such variants of the shift register are not illustrated as they are somewhat outside the scope of the present treatment.

For counting purposes there are two main variants of the shift register which are worthy of our attention. These are the so-called "normal" ring counter and the "twisted" or "switch-tail" ring counter.

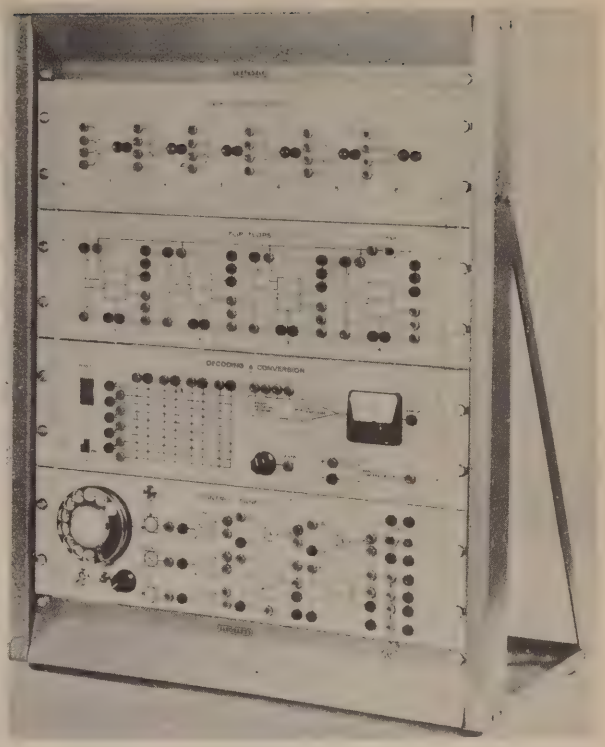
The **normal ring counter** (often simply called the "ring counter") consists of a shift register in which the two ends have been joined to form a closed loop. A pattern of logical values is circulated around the ring or loop by the input pulses, so that at any time it is possible to note the number of pulses which have been received by observing the position of the logical value pattern. Figure 2 shows a small ring counter employing five flip-flops; those shown are of the gated R-S variety, although the J-K type are equally suitable.

It is usual with normal ring counters to circulate a pattern consisting of a single logical "1" together with zeros for all the remaining values; this makes it fairly easy to determine at any time the "position" of the pattern around the ring.

Thus in a typical application of the counter shown in figure 2 the reset-set circuitry would be arranged so that prior to the arrival of input pulses FF1 was set (Y=1) and all four other flip-flops reset. On the arrival of the pulses the "1" would then be shifted in turn to FF2, FF3, FF4 and so on. The fifth pulse would return it to 1 to begin a new cycle.

This can be conveniently shown by the following table. Here, as before, the values in each flip-flop column repre-

Introduced in this chapter is the second panel of the author's digital demonstrator-breadboard unit, shown here complete. The complete unit is described in chapters 12 and 13.



sent the value of the Y output after each input pulse:

INPUT PULSE	FF1	FF2	FF3	FF4	FF5
0	1	0	0	0	0
1	0	1	0	0	0
2	0	0	1	0	0
3	0	0	0	1	0
4	0	0	0	0	1
5	1	0	0	0	0
6	0	1	0	0	0

In passing it is perhaps worth pointing out that the normal ring counter **must** have a preset pattern of values in order to perform counting. If all elements are either set to Y=1 or reset, to Y=0, counting cannot occur simply because there is no pattern to "pass around the ring."

It should be fairly obvious that the normal ring counter will count only in multiples of the numbers of elements in the ring. To count to decimal "N," N flip-flops or other bi-stable elements must be used — unless special arrangements are made, such as cascading ring counters or combining complementing stages and ring counters.

As with ripple-carry counters the normal ring counter may also be used for scaling and frequency division. Thus a ring of five elements becomes a ×5 scaler or frequency divider, etc. As with counting it is possible to extend the range by cascading ring counters or combining them with complementing stages.

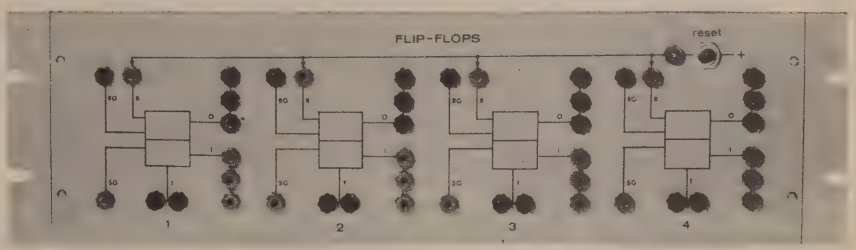
In a normal ring counter circulating a single "1" there are only two elements which change state following any input pulse: the element which switches from 1 to 0, and its successor which switches from 0 to 1. Thus only two elements have to "settle down" following an input pulse before the counter is ready for a following pulse. Because of this the normal ring counter is quite suitable for counting at high speeds.

Even so, however, it is not as suitable as the other counting variant of the shift register: the **"twisted" or "switch-tail" ring counter.** This type involves only **one** element transition per input pulse, and in addition is able to use the same number of elements to count twice the number of pulses.

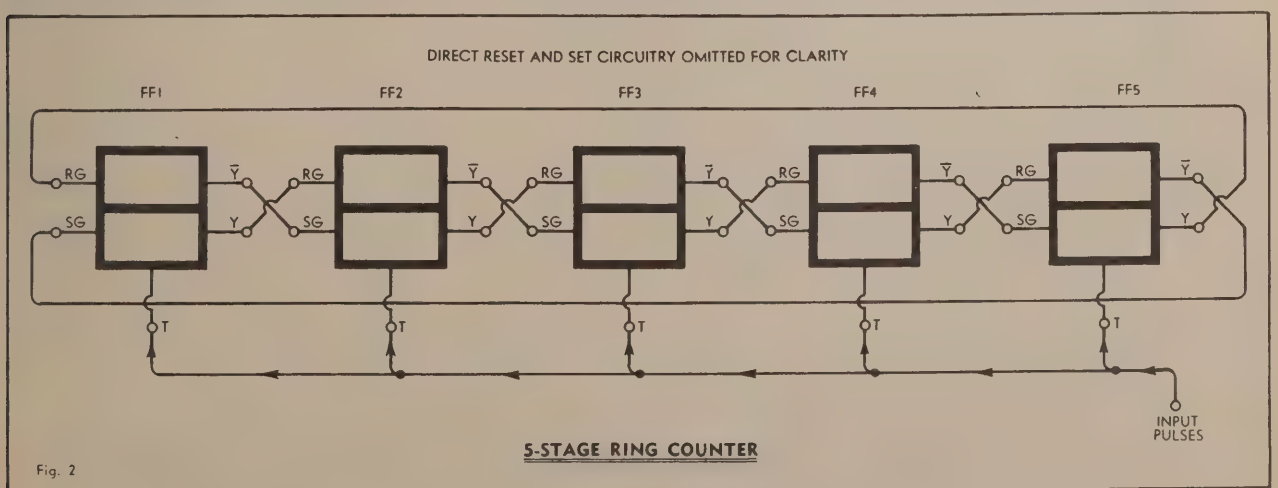
Figure 3 shows the logic diagram for a twisted ring counter involving five elements; here again either gated R-S or J-K types may be used. It may be noted that the only difference between this type and the normal ring counter is that the end-to-end connections are made directly electrically — giving a **logical** twist or inversion.

The twisted ring counter does not have to have to be supplied with a preset pattern as did the normal ring counter: it generates its own pattern, as a result of the logical inversion produced by the twisted end-to-end connections. Prior to the arrival of input pulses the elements are simply reset to Y=O as with ripple-carry counting elements.

On the arrival of the first pulse, FF1 sets to Y=1 — because owing to the logically twisted connection it must take a value **opposite** to that of FF5. Similarly when the second pulse arrives FF1 remains set, because it is still forced to do so by FF5; however, in addition FF2

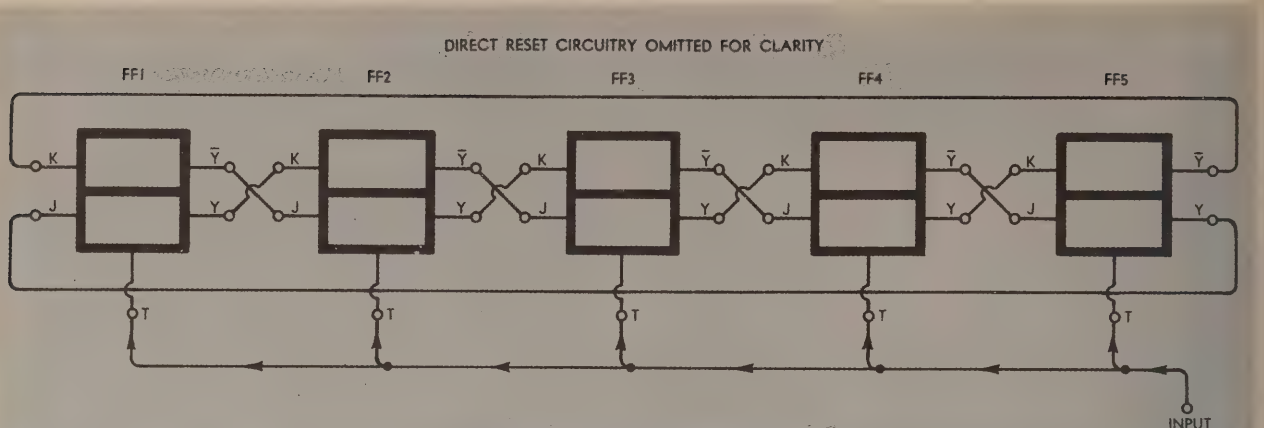


A close-up of the flip-flop panel described later in this article. The four flip-flops may be either the gated R-S or J-K variety, as desired.



5-STAGE RING COUNTER

Fig. 2



5-STAGE TWISTED RING COUNTER

Fig. 3

is set by the second pulse, as FF1 controls its switching.

On the arrival of the third pulse, FF1 remains set; so does FF2. However, FF3 now sets also under the control of FF2. Similarly on the fourth pulse FF1, FF2 and FF3 remain set while FF4 sets; and on the fifth pulse FF5 sets while FF1, FF2, FF3 and FF4 remain set.

On the arrival of the sixth pulse, counting continues but in a different way: now FF1 is forced to **reset** because this time FF5 is set. And on the seventh pulse FF2 also resets under the influence of FF1, which remains reset because FF5 in turn remains set. Thus for the second phase of operation the elements of the register progressively and cumulatively reset, until on the tenth pulse all five are reset and ready for a new cycle of operation.

This operation is conveniently shown by the following table, where again the values shown in the columns represent the values at the flip-flop Y outputs:

INPUT PULSE	FF1	FF2	FF3	FF4	FF5
0	0	0	0	0	0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	1	0
5	1	1	1	1	1
6	0	1	1	1	1
7	0	0	1	1	1
8	0	0	0	1	1
9	0	0	0	0	1
10	0	0	0	0	0
11	1	0	0	0	0
12	1	1	0	0	0

It may be seen from the table that not only does the twisted ring counter involve only **one** element transition per input pulse, making it ideal for high-speed counting, but it also provides an effective count of twice that provided by the normal ring counter. Expressed generally, a twisted ring counter of "N" elements provides unique value-patterns for a maximum of 2N input pulses.

Readout of the count of both normal and twisted ring counters is performed in much the same way as with the ripple-carry counter. Low-loading indication circuitry is connected either directly or via decoding circuitry to the outputs of the bi-stable elements in order to sense their state. As mentioned earlier, readout and decoding techniques will be discussed in later articles.

Thus far in this discussion of bi-stable counting circuitry, we have considered what might be termed "straight" counting, where the bi-stable elements count in a straightforward manner determined by the counting method adopted and the number of elements involved. Thus a register of "N" elements will count up to one less that the Nth exponent of 2 in pure binary if it is arranged for ripple-carry counting, or up to decimal N or 2N in more-or-less decimal fashion if it is arranged for normal or twisted ring counting respectively.

However, as mentioned in the third article in this series, it is often desired that counting and other operations be performed in a way which allows read-out in a numerical code — usually a BCD code.

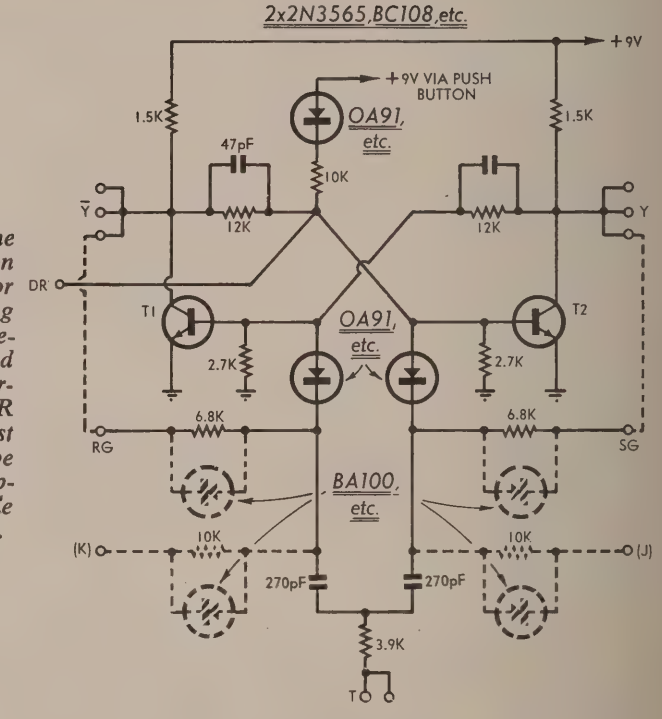
In general, shift counters do not lend themselves to BCD counting; however, this is rarely an embarrassment as in most cases they can be used either for direct decimal counting or for counting in a convenient special code. Perceptive readers will have already realised that a five-element twisted ring counter is ideal for decimal counting; similarly a ten-element normal ring counter can be used for the same purpose, although the latter would be rather uneconomical because of the number of elements employed.

Usually a more convenient way of using the normal ring counter is to combine a five-element register with a single × 2 complementing element. Depending upon whether the single element is placed "before" or "after" the ring counter, this arrangement counts decimal-wise in either "qui-binary" or "bi-quinary" code respectively. It is also possible to count in the latter codes using less than six elements — by combining complementing elements with small twisted-ring counters or gated-carry registers. However such techniques are outside the scope of this article.

As one might expect, it is ripple-carry counters which are mainly associated with counting in BCD codes — because of the basically binary nature of ripple-carry counting

There are many ways in which a ripple-carry counting register can be arranged to count in a BCD code. However, the simplest method is that in which pulse feedback is used to modify

The circuit of the flip-flops used on the demonstrator panel, showing variations between the gated R-S and J-K versions. The NOR gates on the first panel may be used with the flip-flops for simple binary readout.



PANEL FLIP-FLOP
(MAY BE MADE EITHER GATED R-S OR J-K ELEMENTS)

Fig. 5

carry-over at various stages of the count.

This is illustrated in figure 4. Here a register of four gated R-S or J-K elements is shown modified to count in 2421 BCD code by means of two small feedback capacitors connected from the Y-complement output of FF4 to the direct reset inputs of FF2 and FF3.

After the elements are "cleared" or reset via a logical 1 (=+9V) applied to the direct reset line, the operation of this circuit is identical with that of last month's figure 6 up to and including the arrival of the seventh pulse. After the arrival of this pulse, FF1, FF2 and FF3 are all left in the set or Y=1 state, as with the earlier circuit.

However, if it is simply desired that the register count in decimals without any particular preference for the code adopted, any one of these many feedback configurations may be employed—although often the configuration shown in figure 4 will prove to be one of the simplest and hence one of the most attractive economically.

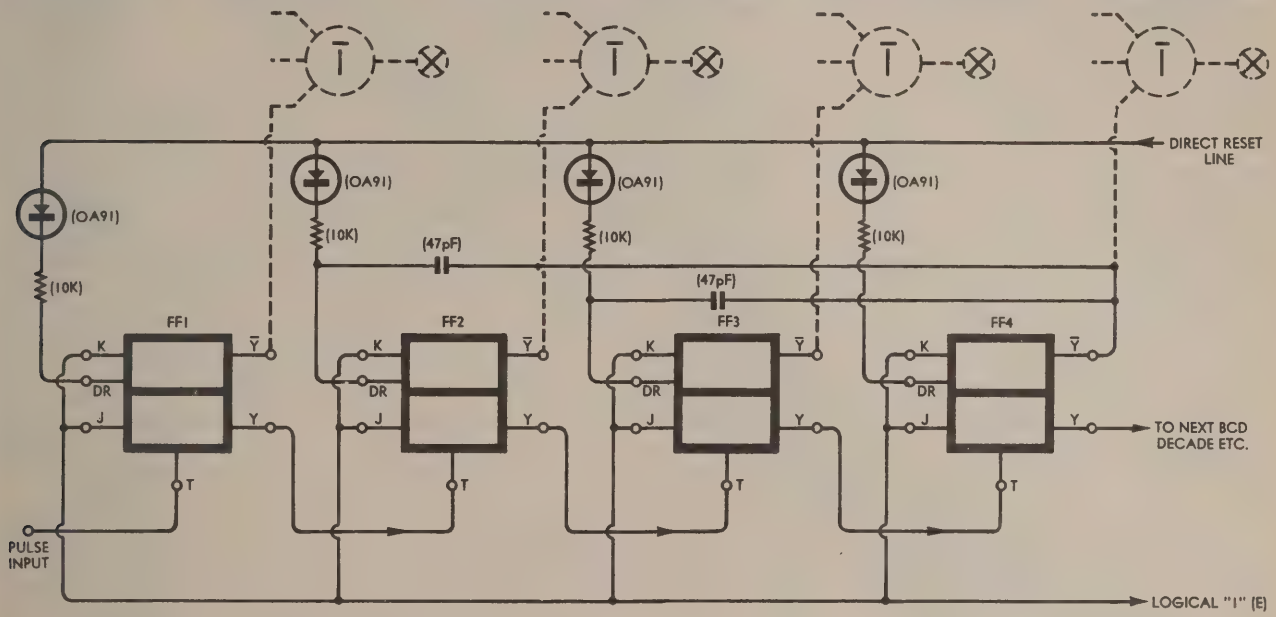
It is worthwhile to realise that as four bi-stable elements are potentially capable of producing 16 different value combinations, adapting them to reset after only 10 of these combinations — regardless of the coding or method employed—amounts to adding a "filler" of 6 to the number being counted.

Just where the filler is added to the count will of course depend upon the code employed; in the circuit of figure 4, it is added at the eighth pulse. This can be seen by noting that in 2421 BCD decimals 8 and 9 are equivalent to decimals 14 and 15 in pure binary. Other BCD codes insert the filler in different places—sometimes by "instalments" rather than all at once.

Other methods of arranging ripple-carry counters to count in BCD codes include gating methods, wherein AND or OR gates operated by the outputs of the elements modify the inter-element coupling at various stages of the count. Such methods are often used with J-K elements, as the "inbuilt" input gating circuitry of these elements can often be arranged to perform the required gating operations.

To conclude the present article, we give the following details of the flip-flop panel shown in the photographs, for the benefit of those readers who wish to construct one for themselves. Such a panel may be used individually for experimental or tuitional purposes, or one or more panels may be combined with the NOR gate panel described in the second article, and with panels to be described in later articles, to form a comprehensive digital demonstrator/breadboard unit.

The basic flip-flop panel features four simple flop-flops which may be made as either gated R-S elements or J-K ele-



J-K RIPPLE-CARRY REGISTER

WITH FEEDBACK FOR BCD COUNTING — ONE DECADE

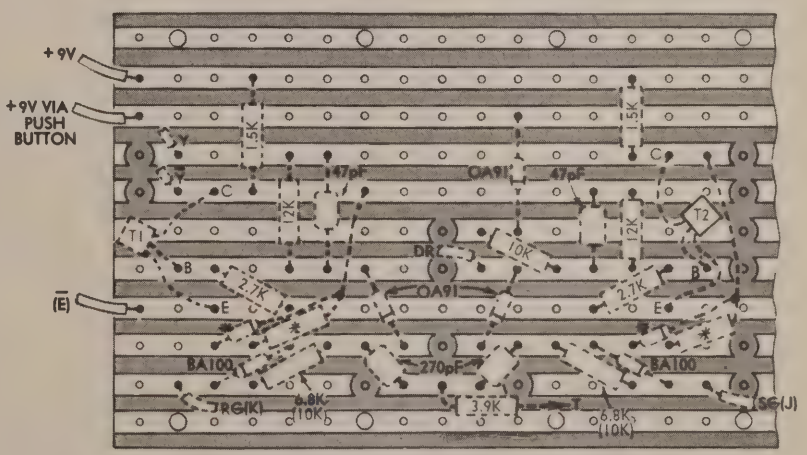
Fig. 4

On the arrival of the eighth pulse, however, the effect of the capacitors becomes apparent. As before, the pulse resets FF1—which in turn resets FF2, which in turn resets FF3, which in turn feeds a pulse to FF4. And as before, FF4 switches over to the set state. However, as this produces a negative-going transition at the Y-complement output of FF4 a pulse is fed via the feedback capacitors to FF2 and FF3 — **with the result that both these elements are promptly set to Y=1 again.**

Note that the direct reset or DR inputs may be used for pulse setting of the elements, by arranging for the pulses to be negative-going. The pulses thus have the opposite effect to the DC signal applied to the direct reset line used to clear the register, in this case setting the elements concerned rather than resetting them.

After the eighth pulse, therefore, the register shows not the pure binary version of decimal 8 or "1000" (reading the Y outputs from right to left), but "1110" — which is the representation of decimal 8 in 2421 BCD code. And with the ninth pulse FF1 is also set, to give "1111" or decimal 9 in the same code. On the tenth pulse all four elements reset to begin a new count.

In a similar fashion, feedback may be used to arrange a four element register to count in many of the BCD codes which were discussed in the third article.



* REQUIRED IF MADE AS J-K ELEMENT

PANEL FLIP-FLOP — WIRING DIAGRAM

Fig. 6


It won't work.
There's a lot more to making modules than buying a few IC's and slapping them on a board. For one thing, the modules almost never work – the first time. Almost never the second time. Occasionally the third time. And that's only the prototype.
Chances are, Digital already has the optimum design, computer tested, fully debugged, manufactured, and sitting there on the shelf. Frustrating, isn't it?
Digital's M Series is the most complete, fully compatible, high speed, integrated circuit, inexpensive line of modules anywhere. We manufacture several million a year – many for our own computers – and know how.
Send for our new Logic Handbook. It tells you what, why, and how to build logic systems from modules. But, alas, not how to build the modules themselves.
Digital Equipment Australia Pty. Ltd.
75 Alexander Street, Crows Nest, N.S.W. 2065
Offices in Melbourne, Sydney, Perth and Brisbane
digital
MODULES · COMPUTERS

ments as desired. (The simple R-S flip-flop may be demonstrated using two of the NOR gates on the panel previously described.) The circuit for each of the flip-flops is shown in figure 5.

As may be seen, the circuit is basically that of figures 4 or 5 discussed in article 4. If gated R-S elements are desired, the circuit is wired virtually to figure 4 of last month; if J-K elements are desired the 6.8K gating resistors are taken to the outputs and additional gating components taken to the front panel terminals—which are then named "J" and "K" rather than "SG" and "RG".

The flip-flops are wired on a length of "Vero-board," as with the NOR gates previously described. The strip of board measures 14¼in x 2¼in, having nine useful conductors with holes on 0.2in centres; it is cut from a stock Vero-board panel coded 4/1001 (one of these stock panels will provide both the NOR gate and flip-flop strips). The conductors are cut and the components wired as shown in figure 6.

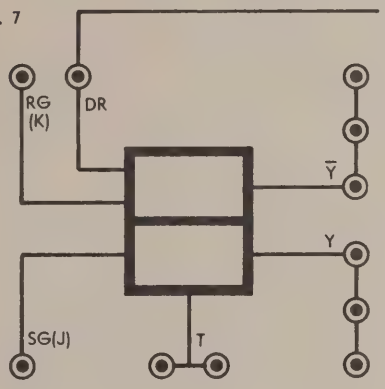
The wiring panel is mounted to the front panel in the same manner as with the NOR gate strip—using small aluminium "U" brackets attached to the rear of the front panel by some of the connection jacks. The latter are of the small "test point" banana-jack variety.

The front panel is from the "Lektrokit" range as before (code number LK-401). Each flip-flop is represented on the front panel by a symbol and set of jacks as shown in the photographs and the diagram of figure 7. The gating terminals are marked either "SG-RG" or "J-K" depending upon the version being used. Note that the "T" input and output terminals are given multiple jacks to facilitate interconnections between elements.

The flip-flop panel may be operated from the same simple power supply described with the NOR gate panel. Thus, the two panels may be used to demonstrate most of the techniques described in this article, with four of the NOR gates used as readout indicators for the flip-flops (connect an input of the NOR gates to the Y-complement outputs).

Note that when using this set-up to demonstrate a normal ring counter it will be necessary to first set all flip-flops and then reset all but one of them to provide the "1" to be circulated. This MUST NOT be done simply by connecting the DR terminal of the flip-flop concerned directly to +9V—if this is done, the transistors will be damaged, due to the absence of protective resistance. Perform the resetting rather by using a 10K series resistor to momentarily connect each DR terminal to the +9V supply line.

Fig. 7



PANEL FLIP-FLOP REPRESENTATION

As may be seen from the photographs, the two prototype panels thus far described in these articles have been mounted in a small Lektrokit rack (code LKR-5011). The rack provides a convient and attractive mounting system for the panels and provides space for the two further panels which are described in further articles.

It should, perhaps, be repeated that both Vero-board and Lektrokit components are obtainable from parts suppliers on order from the importers, E.M.I. (Australia) Ltd. In N.S.W., orders may be placed with the State distributor, Watkin Wynne Pty. Ltd., P.O. Box 318, Crow's Nest.

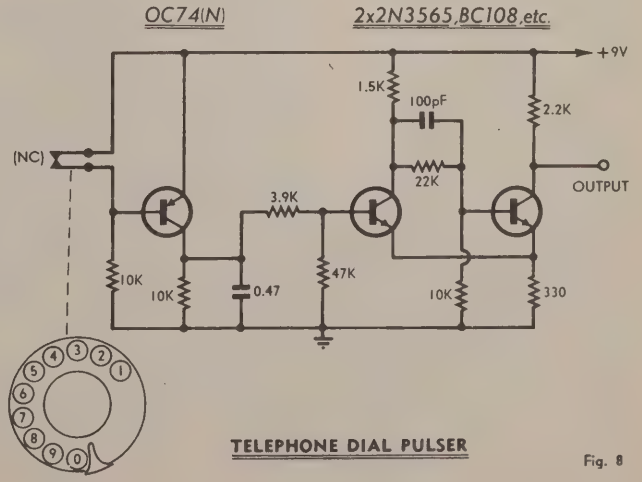
In order that readers may be able to feed pulses into the flip-flops at this stage, we publish the circuit of a decimal pulse generator using a disposals telephone dial (figure 8). As this circuit will be discussed in a later article in connection with the description of one of the later panels, it will not be discussed in detail here. However, the following brief explanation may be useful.

The pulsing contacts on a telephone dial are normally closed, opening for the duration of the pulses. These are used to hold the OC74 transistor off normally so that when they open during a pulse the transistor bottoms (saturates) and the supply voltage appears across the 10K collector load.

The purpose of the 0.47uF capacitor is to partially integrate the resultant pulses. Partial integration is necessary as the dial contacts—like most mechanical contacts—are prone to generate short spurious pulses due to bounce. With partial integration the spurious pulses are "lumped" in with the main pulse and rendered incapable of upsetting or confusing counter operation.

As the partially integrated pulses are somewhat rounded, it is necessary to "square them up" before they are suitable for feeding to the flip-flops. This is done by the circuitry asociated with the two BC108 transistors, which may be recognised as a transistorised Schmitt trigger circuit.

The circuit for a simple pulse generator using a disposals telephone dial. A similar circuit forms part of the fourth demonstrator panel.



TELEPHONE DIAL PULSER

Fig. 8

Chapter 6

A survey of READOUT DEVICES

Introduction — direct inclusion of lamps in the circuit — driver amplifiers — "columnar" vs. "in-line" display — the edge-lit numeral display unit — projection-type units — cathode-ray indicators — miniature neon tubes — numerical neon indicators—the Mullard "Digitube"—the decade trigger indicator — the electroluminescent display panel — the moving-coil numerical indicator.

As we have seen from the earlier chapters, digital circuitry often deals with numerical information. It uses pairs of magnitude and/or polarity values of electrical voltages or currents to represent the numerical information statements, according to arbitrary "logic conventions." And it effectively performs mathematical and logical operations on the information by performing analogous electrical operations on their representations.

Although digital circuitry thus deals with information very effectively as electrical signals, it is not normally the case that human beings can handle the information conveniently or efficiently in the same form. We humans usually feel more confident and work somewhat more effectively with the information presented in the form of printed figures, glowing numerals, indicating pointers, and the like.

In view of this, it becomes essential to provide digital equipment with information **transformation** facilities both at its "input" and at its "output." By means of these facilities the incoming information may be transformed into an electrical representation at the input, and the outgoing information at the output transformed back into a form permitting convenient interpretation by the human user. Input-output transformation is thus an important function in digital equipment.

For convenience, the discussion in this book of the general topic of input-output transformation will be split into three aspects. The first of these concerns readout devices, which are the subject of the present chapter. The second concerns binary- and BCD-decimal decoding, which forms the subject of the following chapter. And the third aspect concerns digital-to-analog and analog-to-digital conversion, which is discussed in chapter 8.

It is not proposed to discuss in the present chapter devices or techniques of the type used in data-processing applications such as punched-card or tape equipment or high speed printers. These are discussed in a later chapter; for the present we will be concerned mainly with devices and techniques associated with digital measuring instruments. And in particular, with readout devices employed in displaying the information stored in transistorised or integrated logic and counting circuitry.

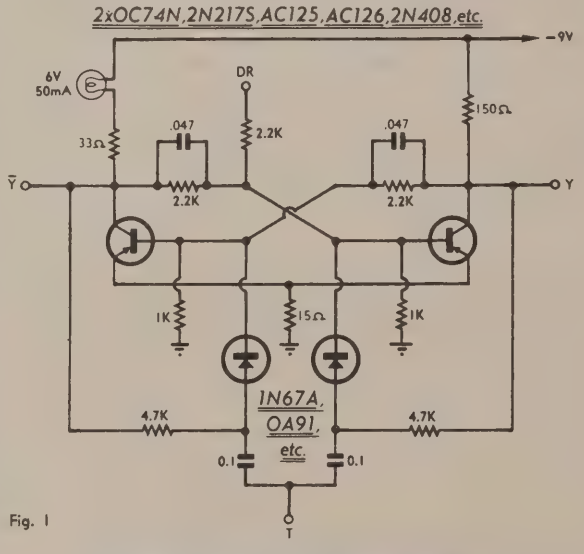


Fig. 1

GATED R-S FLIP-FLOP WITH INDICATING LAMP

To begin, then, probably the simplest way of performing readout with transistorised counting circuitry is to fit low-power incandescent lamps directly in series with the appropriate collector leads. Thus a counting register may use flip-flops of the type shown in figure 1 to give a direct readout of its registration via the 6V 50mA lamps.

However, while simple and economical, this technique has rather serious shortcomings. Perhaps the most obvious of these is that the information readout must be made in the coding system actually used by the counting circuitry. Unless the circuitry actually counts in decimal, the operator is thus faced with learning to read either pure binary or one of the many BCD codes, or perhaps the bi-quinary or qui-binary code.

From the technical viewpoint, there is the shortcoming that the inclusion of lamps directly into the counting circuitry severely limits the high-frequency performance. In order to operate the lamps directly, the circuitry must generally operate at fairly high current levels; not only this, but the lamps themselves provide a somewhat inappropriate load impedance. Both factors tend to reduce the maximum attainable counting speed.

In short, while direct inclusion of low-power incandescent lamps into the counting circuitry is simple and economical, it is really only suitable for readout of circuitry which is counting at low speeds and in a decimal fashion.

Thus the circuit of figure 1, while quite practical, is really only suitable for use in circuitry of the "normal ring counter" variety, at speeds up to about 10KHz. Additionally, it could be used for demonstration of binary and BCD counting on an experimental basis.

A development from the simple lamp technique overcomes both major disadvantages of the latter by providing each lamp with its own "driver" amplifier, an example of which is shown in figure 2. The use of driver amplifiers both reduces the loading effect of the lamps upon the counting circuitry and also permits the use of higher-power lamps for convenient readout.

No less important, it also allows the readout lamps to give an indication in a code other than actually used by the counting circuitry, by permitting the interposition of decoding circuitry between the counting register and the lamp drivers. (The circuit configurations involved in decoding are discussed in some detail in the next chapter.)

The circuit configuraton shown is that of a simple non-inverting driver suitable for operation of 12V lamps up to 150mA. It is designed for use with

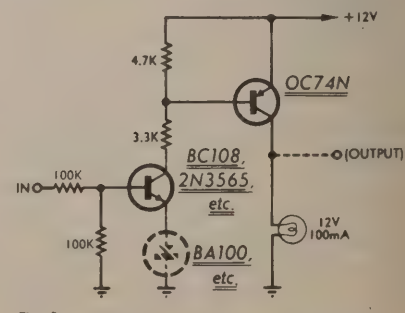
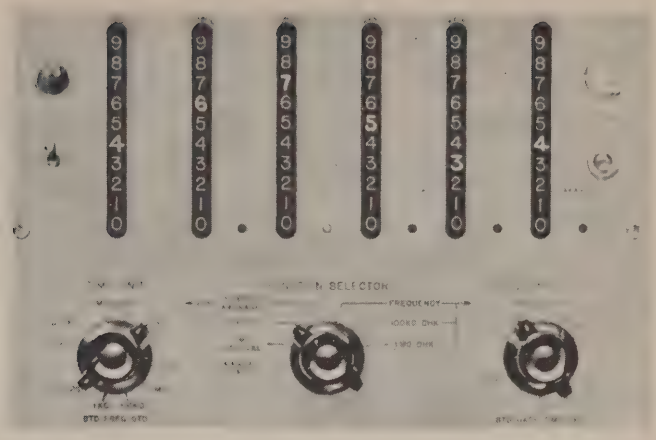


Fig. 2

LV INCANDESCENT LAMP DRIVER



The "columnar" type of numeral display, as used on early digital instruments. (Courtesy Hewlett-Packard Co.).

counting circuitry operating from a +9V supply, for example the flip-flop circuitry discussed in the preceding article. As may be seen, it employs a complementary NPN silicon and PNP germanium transistor pair in cascade, giving a current amplification of about 1000. The transistors used are very low cost types and are easily obtained.

The "fan-in" or effective loading effect is quite low—approximately 90uA. The lamp voltage may be used as output for further logic if desired. The silicon diode shown dashed in series with the NPN emitter is to provide additional cutoff bias where the input "0" level may tend to rise above earth potential. It is mainly necessary when the driver is used with decoding circuitry.

Note that while this circuit performs DC amplification, it is designed solely for digital "on-off" use. It is not suitable for linear amplification.

Normally, if such lamp drivers are employed they would be used in conjunction with decoding circuitry to give decimal readout. Thus in the typical case, each counting decade would have an associated decoding circuit together with a set of 10 driver circuits and 10 lamps signifying the numerals 0-9.

The simplest way of using the 10 lamps for display is to arrange them in a vertical column. They may be mounted either in simple fashion with suitable panel lettering, or alternatively mounted behind transparent numeral engravings so that each lamp illuminates the appropriate numeral.

A "columnar" display of this type is quite practical and in fact was used in a number of early digital instruments. Illustrated is an early "Hewlett-Packard" electronic counter, which used the transparent-numeral columnar system. (The counter concerned actually used valve circuitry and neon lamps, but the picture illustrates well this type of display.)

The disadvantage with a display of this type is that readout tends to be slow and tiresome as a result of the "jagged" nature of the display. Rapid and convenient readout is difficult, as each digit may occupy any one of 10 vertical positions.

From the operator's point of view, the "ideal" display is one wherein the numerals for each digit always appear in the same position, so that the complete registration may be read conventionally as a contiguous horizontal number. For more practical readout systems it is thus desirable to produce such an "in-line" display.

Two main techniques have been developed to permit an in-line display when using the 10-lamp readout system.

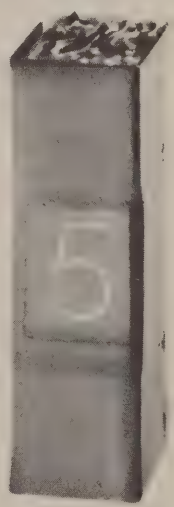
One system uses the lamps to edge-light small sheets of clear plastic, upon which the numerals are engraved in small dots so that, when a lamp lights, the engraved numeral appears to glow. The sheets are stacked one behind the other behind a viewing window, the dot-structure of each engraving permitting even the rearmost numerals to be seen relatively easily when illuminated.

A readout unit employing this method is pictured. It employs a total of 11 plastic sheets, with the appropriate miniature 12V 100mA lamps fitted alternately above and below the stack and wired to small printed-wiring boards. It measures approximately 2½in x ¾in x 1¼in and provides compact display of the numerals 0-9 together with a decimal point.

The alternative 10-lamp in-line display system uses the lamps to directly illuminate microfilm numeral transparencies, which are then projected optically on to a small round-glass screen. As direct lighting is rather more efficient than edge-lighting, this method gives a somewhat brighter display; however, the price tends to be considerably higher because of the precision optical components required.

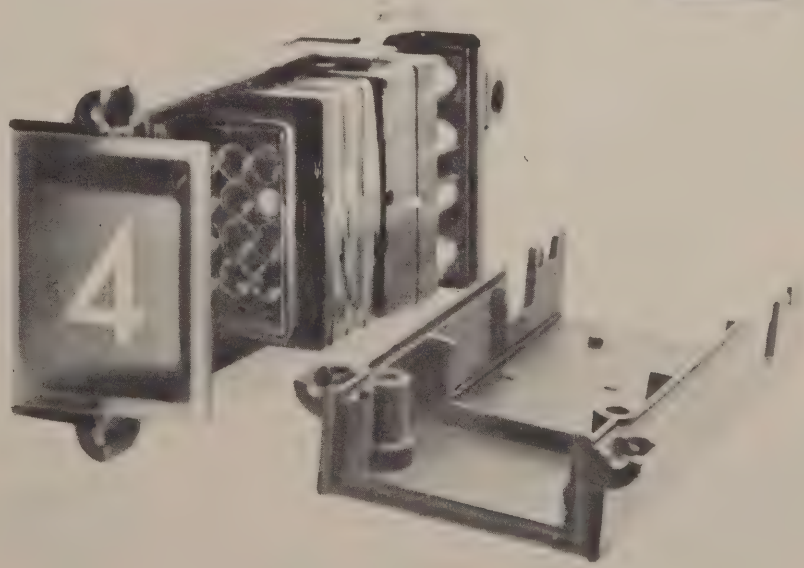
A sample projection-type unit is pictured, with one side cover removed to show the internal construction. The unit has twelve 12V 150mA lamps, which mount in a removable common carrier at the rear. Between the lamps and the ground-glass screen are twelve optical systems, each consisting of a microfilm numeral transparency and the necessary condensing and objective lenses. The unit provides display of the numerals 0-9, together with polarity symbols "+" and "—."

While the low-voltage readout devices and systems discussed thus far are quite practical, and have been and still are used in many digital applications, they



A typical indicator of the edge - lighting type, the Koten RK-E1 by Okaya Musen Co. (Courtesy Craftsman Glass Pty. Ltd).

The projection-type "One-Plane Readout" by Industrial Electronic Engineers, Inc. (Courtesy Craftsman Glass Pty. Ltd.)



VENNER COUNTER-TIMERS FOR TOP VALUE

TSA6634A/2

10 MHz FREQUENCY COUNTER TIMER

FREQUENCY MEASUREMENT
Range: D.C. to 10 MHz
Sensitivity: 75 mV to 250 V
(variable sensitivity control)
PERIOD MEASUREMENT
Range: 10 Hz to 1 MHz
TIMING
Time Units:
1μS to 10 seconds in decade steps.
(Time measurement from 1μS to 10^5 secs.)
Crystal Stability: $\pm 1 \times 10^{-6}$
OPTIONAL EXTRAS
A stored display version is available, type TSA6634A/2M.

The printer option (TSA6634A/2Mz) can only be fitted to the TSA6634A/2M Provides 1248 b.c.d. output
Logic '0'=0V; Logic '1'= +4V

TSA5538/Z

110 MHz DIGITAL FREQUENCY METER
(with stored display)

FREQUENCY MEASUREMENT
Range: D.C. to 110 MHz
PERIOD MEASUREMENT
Range: 10 Hz to 10 MHz
COUNT
Range: D.C. to 110 MHz
TIMING
Time Units.
0.1μS to 10 seconds in decade steps.
(Measures time from 0.1μS to 10^9 secs.)
Internal standard:
5 MHz oven controlled crystal oscillator.
Crystal Stability: $\pm 1/10^8$
Printer Output: 1248 b.c.d.
Logic '0'=0V; Logic '1'= +4V

OPTIONAL EXTRAS
Improved internal frequency standard $\pm 1/10^9$.

TSA6636/2

12.5 MHz FREQUENCY COUNTER TIMER

FREQUENCY MEASUREMENT
Range: D.C. to 12.5 MHz
PERIOD MEASUREMENT
Range: 10 Hz to 1 MHz
COUNT
Range: D.C. to 12.5 MHz
TIMING
Time Units:
1μS to 10 seconds in decade steps.
(Time measurement from 1μS to 10^7 secs.)
Crystal Stability: $\pm 1 \times 10^{-6}$

OPTIONAL EXTRAS
A stored display version is available, type TSA6636/2M

The printer option offered (Type TSA6636/2MZ) can only be fitted to the TSA6636/2M. Provides 1248 b.c.d. output
Logic '0'=0V; Logic '1'= +4V

VENNER **Venner Electronics Ltd.**

H. ROWE & CO PTY. LTD. Sole Australian Agents | Adelaide, Brisbane, Hobart, Melbourne, Perth, Sydney.

are by no means the only devices or systems in use. Many other devices have been developed, some of which are claimed to offer the advantage of brighter and clearer display, while others claim advantages in terms of power supply economy, drive circuit simplicity, reliability, and so on.

Small **cathode-ray indicator tubes** of the miniature "magic eye" variety have been used at various times, both for binary and columnar decimal display. These produce quite a useful display for a low power dissipation, and have a very high input impedance. An example is the Mullard DM160, which produces a bright rectangular glow when operating from supply voltages as low as +50V. The directly heated cathode requires 30mA at 1V, AC or DC.

A typical circuit configuration used with this type of indicator is shown in figure 3.

Here the negative logic convention is used, with 0=0V, 1=−6V. In the absence of input voltage, the silicon PNP transistor is cut off; this connects the grid of the tube to −6V via an effective 110K series resistor, cutting it off also.

To produce an indication, logical 1 must be applied to the input. This drives the transistor to saturation, removing the negative bias from the tube and permitting it to conduct. Removing the input voltage reverses this mechanism, cutting off both transistor and tube as before.

Although quite practical, the cathode-ray indicator is not often used nowadays. Because it relies upon a thermionic cathode it requires an additional power supply compared with alternative devices; apart from this, the useful life tends to be comparatively short. These factors combine with a moderately high costs to make it relatively unattractive compared with alternative devices.

Perhaps the most widely used group of display devices are based on the cold-cathode discharge tube in one form or another.

The simplest such device is the **miniature neon tube,** which like the simple incandescent lamp and the cathode-ray indicator, is a "single-bit" display device and may be used for binary and columnar decimal display. Figure 4 shows one possible way in which such a tube may be used with transistor circuitry.

Here a BF115 silicon NPN transistor is used as an input amplifier and inverter, followed by an AC127 germanium NPN as an emitter follower. The neon tube is an NE-2 or similar, with a striking voltage around 65V and a maintaining voltage of approximately 50V. The circuit operates as follows.

With a logical 0 (=0V) applied to the input, the first transistor is cut off. Its collector thus approaches +30V, carrying with it the base of the second transistor. The latter, therefore, conducts and applies a similar voltage across the 10K resistor in series with the neon tube.

Because the other side of the tube connects to a +70V supply, this reduces the total effective tube voltage to only (70−30)=40V, which is insufficient to cause the tube to ignite. Thus logical 0 results in the tube remaining dark.

If on the other hand logical 1 (= +9V) is applied to the input of the circuit, the first transistor conducts to saturation. This means that the collector voltage drops to almost zero, carrying with it the base of the second transistor.

The second transistor is thereupon cut off, removing the +30V from the 10K resistor. And with the voltage removed from the resistor, the neon tube receives the full +70V supply voltage. It therefore ignites to give the desired indication.

While giving a fairly bright indication for a modest power consumption, the simple neon tube indicator suffers from the same disadvantage as did the incandescent lamp used in the circuit of figure 2, in that alone it is really only suitable for the columnar type of decimal display.

Here edge-lighting and projection techniques for producing numeral display are hardly feasible in view of the modest light output available.

As some readers may be aware, however, there is a development from the simple neon tube which was produced specifically for numeral display. This is the **numerical neon indicator tube,** marketed under such trade names as "Nixie" (Burroughs Corp.), "Digitron" (Ericsson Telephones Ltd.) and "Numicator" (Hivac Ltd.). Currently such tubes are the most commonly used of all digital readout devices, by virtue of their bright, easily interpreted display, their high efficiency and excellent reliability.

There are two basic variants of the numerical indicator tube, and the construction of both may be seen from the photographs. In one the numerals are viewed through the side of the tube envelope; in the other they are viewed through the end. Each method of construction has certain advantages with regard to numeral size, suitability for close mounting of tubes, and so on, giving the two somewhat different areas of application.

In both cases the tube is basically a multi-cathode neon tube in which the cathodes are either formed from wire or stamped from thin metal sheet in the shape of the desired numerals or mathematical symbols. The common anode is a fine wire gauze or perforated sheet through which the cathodes may be observed.

When a selected cathode is made negative with respect to the anode by about 180VDC, it glows with the familiar pink neon aura. The aura extends around the shaped cathode to a sufficient extent to permit easy observation even when the selected cathode is rearmost in the array. The unwanted cathodes are either left "floating" electrically or are clamped at a voltage above that of the maintaining voltage of the active cathode — e.g., at 100VDC negative with respect to the anode.

Figure 5 shows two typical ways of driving such tubes. Electrically, the

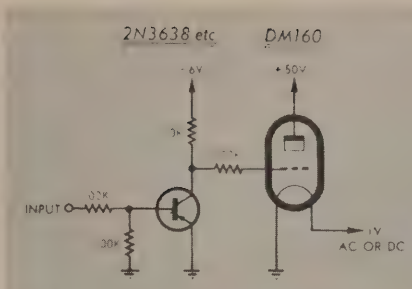


Fig. 3 CATHODE-RAY INDICATOR

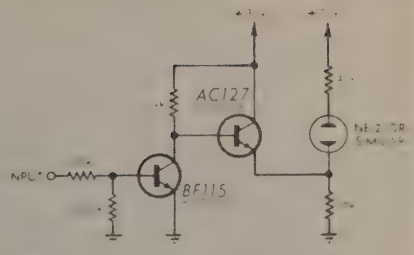


Fig. 4 NEON TUBE INDICATOR

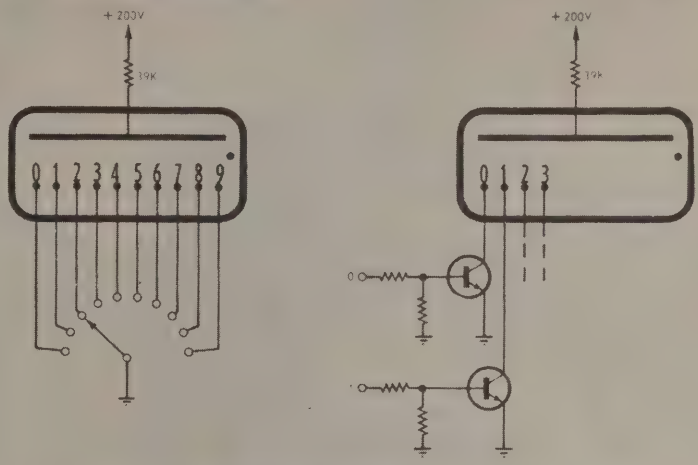
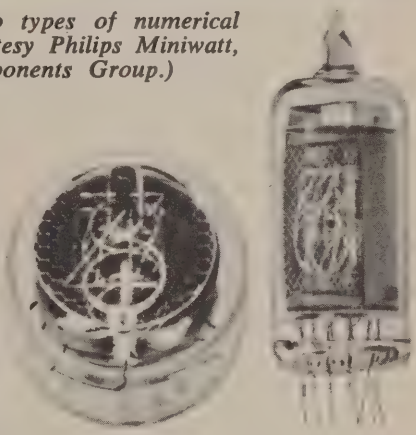
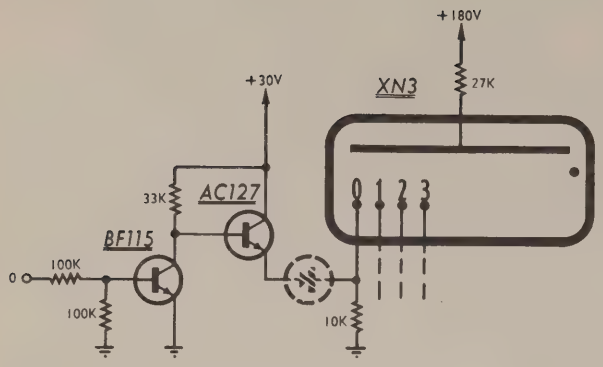


Fig. 5 NUMERICAL NEON INDICATOR

Examples of the two types of numerical neon indicator. (Courtesy Philips Miniwatt, Plessey Components Group.)





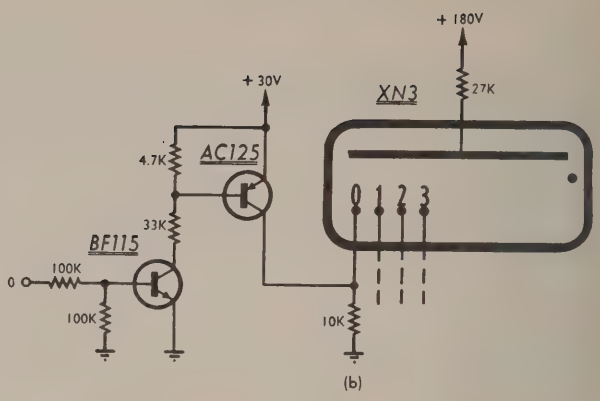


Fig. 6

simplest method of all is that shown at the left, using a rotary switch to connect the desired cathode to the negative line. However, such a system does not usually lend itself to convenient operation from electronic counting circuitry.

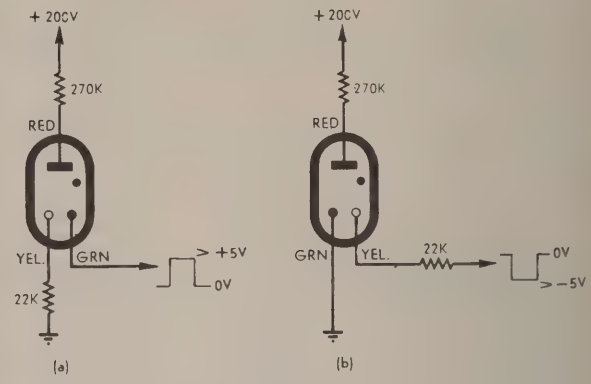
A more convenient method is to use transistors as switches between each cathode and negative, as shown on the right of figure 5. Here indication of the desired numeral is achieved simply by applying a suitable bias to the base of the appropriate transistor, to turn it "on."

The disadvantage of the latter type of circuit is that it requires the use of fairly costly high-voltage silicon transistors. Silicon transistors are required because it is essential that the switches have a low leakage when intended to be "off"—otherwise other numerals will glow along with that selected. And transistors with a high BVceo voltage rating are required, because in operation the unselected cathodes "float" at a potential which in some cases becomes an appreciable proportion of the anode potential. (**BVceo** is the collector-emitter breakdown voltage with zero base current.)

The circuit configuration of figure 4 can be adapted to obviate this problem, by allowing the tube to be driven using economical lower voltage transistors. This is illustrated in figure 6(a).

Here as before a pair of transistors is arranged so that in the absence of input voltage (or strictly, with logical 0 input) a +30V cutoff bias is applied across a resistor in series with the tube cathode. Alternatively when a logical 1 is applied to the appropriate input, the transistors saturate and cutoff respectively, removing the bias from the cathode and allowing it to take the discharge.

The diodes shown dashed in series with the AC127 emitters are only required if



TG121A "DIGITUBE" Fig. 7

the tube is operated at current levels sufficient to produce a voltage across the cathode resistors in the "on" state of greater than (30V + BVebo), where **BVebo** is the reverse breakdown voltage of the AC127 base-emitter junction. In such (rare) cases the diodes prevent possible transistor damage. With the economy XN3 tube and the circuit values shown, the diodes are not required.

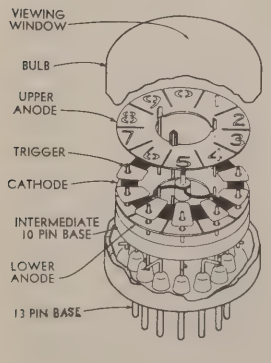
Where inverting or negative logic drive is required, the circuit configuration of figure 6(b) may be used. This is very similar to that just discussed, except that the second transistor is an AC125 or similar PNP type used as a common-emitter amplifier. Here the logic convention is the opposite to that of the other circuit, with logical 0 = +9V and 1 = 0V. To make a given cathode glow, the appropriate input terminal is grounded.

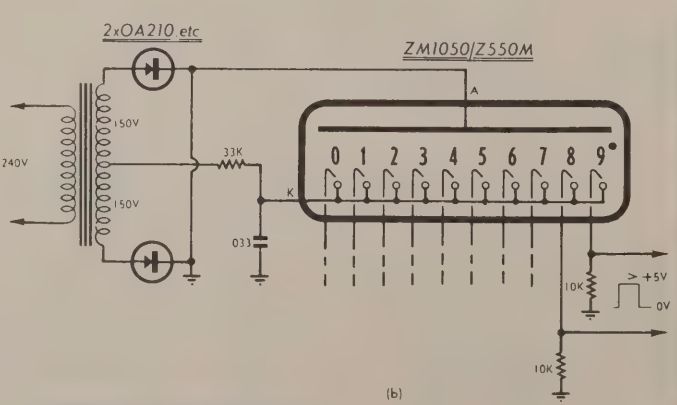
Drive circuitry for numerical indicator tubes need not necessarily be transistorised—although the major proportion of such circuitry is, in fact, of this type, because of the compatibility with transistorised and integrated counting/logic circuitry. Alternative switching devices include thermionic valves, three and four-element neon trigger tubes, thyristors or "SCR's," and other controlled semiconductor switches. Numerical indicator tubes may also be used to display the registration of cold-cathode counting tubes and vacuum trochotrons.

The simple neon tube and the numerical indicator tube are discharge tubes which are not in themselves particularly suited for the small drive voltages available from semiconductor counting or logic circuitry. Thus their use entails the employment of somewhat elaborate drive circuitry.

There are, however, cold-cathode display tubes which have been designed specifically for use with low-voltage semiconductor circuitry. Two such devices will now be discussed.

The **"Digitube"** (Mullard Ltd.) is a development from the simple neon indi-





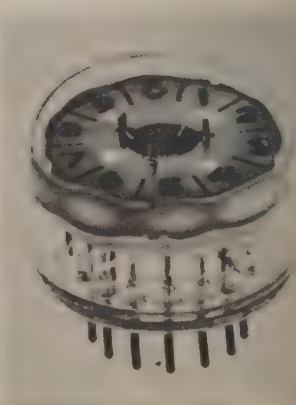


Fig 8

NUMERICAL TRIGGER INDICATOR

cator tube, which it closely resembles. The difference between the two is that instead of a single cathode the Digitube has two—one of which is obscured behind a mask.

The geometry of the tube construction is such that if both cathodes are at the some potential with respect to the anode, the **obscured** cathode virtually captures the total discharge. Under these conditions, the visible cathode remains dark, giving the "0" indication.

To produce a visible glow for indication of "1", the discharge is transferred to the visible cathode simply by making this electrode a few volts negative with respect to the obscured cathode. As shown in figure 7, this may be achieved by feeding either a positive signal to the obscured cathode (green wire) or a negative signal to the visible cathode (yellow wire).

In either case the drive signal required is but a few volts, and is easily supplied by most transistorised circuitry. The loading is almost negligible, making for very simple drive circuitry. And on the removal of the drive signal, the tube discharge returns automatically to the abscured cathode, providing the tube-current is maintained within specified limits.

Fairly obviously, the Digitube is a single-bit display device with applications similar to those of the incandescent lamp and simple neon tube. However, the second low-voltage-drive discharge tube to be discussed is a numerical indicator —although not of the "in-line" variety. This is a development of the neon trigger tube which may for convenience be called the **"Decade Trigger Indicator."**

An indicator tube of this type is the Mullard - Amperex ZM1050 / Z550M, whose construction may be seen from the photograph and from the diagram of figure 8(a).

The tube is an end-viewing device having 10 sector cathodes connected together via a metal ring, with ring-shaped anodes both above and below. The anode ring nearest the viewing window is provided with numeral cutouts, while each cathode sector has a hole through which protrudes a small trigger electrode. All 10 trigger electrodes are brought out separately, together with the common anode and cathode leads.

As may be seen from the circuit diagram of figure 8(b) the tube is operated from a power supply employing a full wave rectifier—but essentially **lacking filtering.** The tube anode is earthed, with the supply negative and the cathode floating. All trigger electrodes are taken to earth via suitable resistors, across which the drive signals are applied.

Operation is as follows: Because of the unfiltered supply, the potential at the tube cathode drops to zero periodically at twice the mains supply frequency (i.e., 100Hz). Each time this occurs the tube must necessarily extinguish. During each mains half-cycle, the cathode potential rises (negatively) and falls sinoidally.

As the cathode potential becomes more and more negative the potential across the tube approaches the ignition point. And it is with a trigger electrode that the discharge will first occur, because these electrodes are physically nearer the cathode than is the anode.

It should be fairly obvious that if any one of the trigger electrodes is made slightly positive with respect to the rest, it will be with **this** electrode that the discharge will occur. And the construction of the tube is such that, if the ionisation produced by this initial discharge is sufficient, the main anode takes over the discharge almost immediately.

Numerical neon tube display is probably the most often used in modern digital instruments, offering bright and easily-read display at a modest power consumption. Illustrated is a modern counter-voltmeter by Hewlett-Packard. (Courtesy Hewlett-Packard Aust. Pty. Ltd.)

Providing the current levels are kept within a certain range, this produces a cathode aura confined to the associated cathode sector, and visible through one of the numeral cutouts in the front anode ring. Thus the tube displays the numeral concerned.

The tube can thus be arranged to display any of the 10 numerals simply by applying a small positive voltage to the appropriate trigger electrode. And as the discharge extinguishes at the end of every mains half-cycle, the display may be changed simply by removing the drive voltage from one trigger electrode and applying it to another. The tube is thus eminently suitable for use with low-voltage semiconductor circuitry.

It may be noted in passing that with suitably designed circuitry the decade trigger indicator may also be used as a counter and scaler. Used in this fashion it will operate at speeds up to about 1000Hz.

To conclude this brief survey of digital readout devices and techniques mention should be made of two types of device which are currently encountered rather less frequently than the types so far discussed. The first of those is the **electroluminescent display panel,** a relatively recent development which may well become very popular in the future.

As the name suggests, the operation of the device depends upon "electroluminescence," the phenomenon whereby certain compounds called **phosphors** emit visible light when subjected to an electric field.

An electroluminescent panel is basically a capacitor in which a phosphor powder is supported within a plastic or ceramic dielectric which is sandwiched between transparent conductive electrodes. An alternating voltage of about 200V RMS at 400Hz is applied between the two electrodes, producing a corresponding electric field in the dielectric and phosphor. The resulting glow is viewed through one of the transparent electrodes.

For digital readout purposes panels are arranged to have a number of electrically isolated segments, as in figure 9. Then by applying voltage across various combinations of the segments, the panel may be made to display representations of numerals, alphabetical figures, or mathematical symbols.

Electroluminescent readout panels are electrically efficient, take up little space and may be inspected through a wide viewing angle as a result of the effective "area source" of light. In addition they promise to have extremely high reliability, in view of the lack of heated filaments and vacuum or gas filling.

Drive circuitry tends to be somewhat more complex than with other readout devices, as apart from the usual binary or BCD-to-decimal decoding the circuitry must also perform **"translation"** to ensure that the appropriate panel segments are energised for each numeral displayed. However, panels have recently been developed which perform "internal" translation using electroluminescent layers, optical translation masks and photoresistive layers bonded to the rear of the panel proper.

The final readout device to claim our present attention is the **moving-coil numerical indicator,** of which the "Digivisor" unit (E.A.C. Ltd.) is the best-known example. Designed primarily for low cost applications, this type of device is not widely used on account of the relatively lower reliability of a moving mechanical device compared with a fully electronic system.

It is discussed here primarily for the sake of completeness. But also because the drive arrangement employed provides an excellent introduction to the concept of **digital-to-analog conversion,** which will be discussed in some detail in chapter 8.

As may be seen from figure 10 the construction of this type of device resembles that of a conventional moving-coil microammeter. The difference is that in place of a pointer and dial scale, the numerical indicator has a small microfilm transparency which carries the numerals or symbols to be displayed. A small lamp and optical system projects an image of one of the numerals on to a finely ground glass viewing screen; which numeral is projected and displayed depends upon the deflection of the moving coil, which in turn depends upon the coil current.

Fairly obviously, this device requires a drive signal which is not in the form of a set of "on-off" signals, but rather in the form of a current capable of being given ten amplitude values to correspond to the various numerals. In other words, it requires a many-valued **analog** drive signal rather than a two-valued **digital** signal.

To operate the device from a digital counting register or logic circuit thus requires drive circuitry capable of performing a process of digital-to-analog conversion.

Digital-to-analog conversion and the complementary process of analog-to-

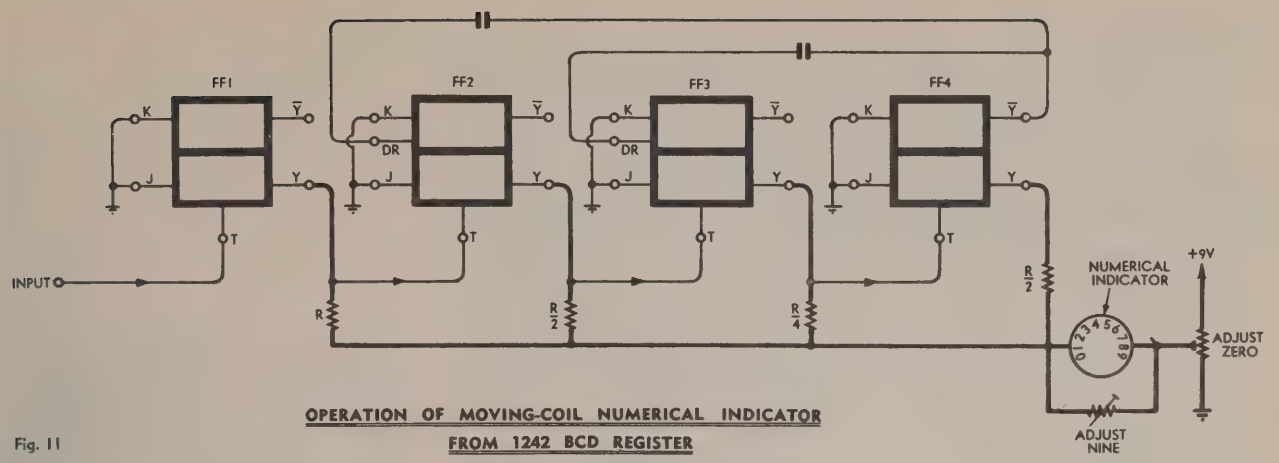


Fig. 11 **OPERATION OF MOVING-COIL NUMERICAL INDICATOR FROM 1242 BCD REGISTER**

digital conversion will be discussed in some detail in the next article in this series, along with the processes of encoding and decoding. However, the following brief discussion may serve here both as a rounding-off to the description of the moving-coil numerical indicator and as a preliminary insight into the concepts to be treated next month.

Figure 11 shows the basic circuitry used when the moving-coil indicator is used with a bi-stable register counting in 1242 BCD code. It consists of a resistive summing network, in which the values of the summing resistors are inversely proportional to the **binary weights** of the counting stages—i.e., the first has a value "R," the second R/2, the third R/4, and the fourth R/2. The resistors connect from the Y output terminals of the bi-stable stages through the indicator to a variable low voltage.

The value R is arranged to be considerably larger than the internal resistance of the indicator coil. Thus when any of the bi-stable elements sets to Y=1, a current flows through the associated resistor and the indicator which is effectively determined solely by the value of the resistor.

As the resistor values are inversely proportional to the binary weightings of the stages, this means that the setting to Y=1 of each stage produces an indicator current **directly proportional** to its appropriate binary weighting. Thus the setting of FF1 produces 1 unit of current; that of FF2, 2 units of current; that of FF3, 4 units of current; and that of FF4, 2 units of current.

Hence any combination of values in the four stages produces a corresponding analog current value through the indicator. If FF1 and FF2 are both set, 3 units of current flow, and so on.

The adjustable shunt across the indicator coil is simply for the purpose of calibrating the "9" indication. Hence in a practical case the indicator may have a basic sensitivity of 250uA, while the value R is chosen to give unit current steps of 30uA; the shunt is then used to accurately set the effective sensitivity at 270uA for a "9" reading.

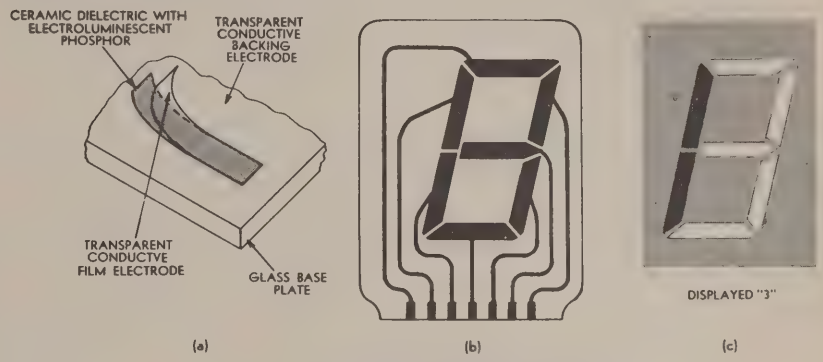


Fig. 9 **ELECTROLUMINESCENT INDICATOR**

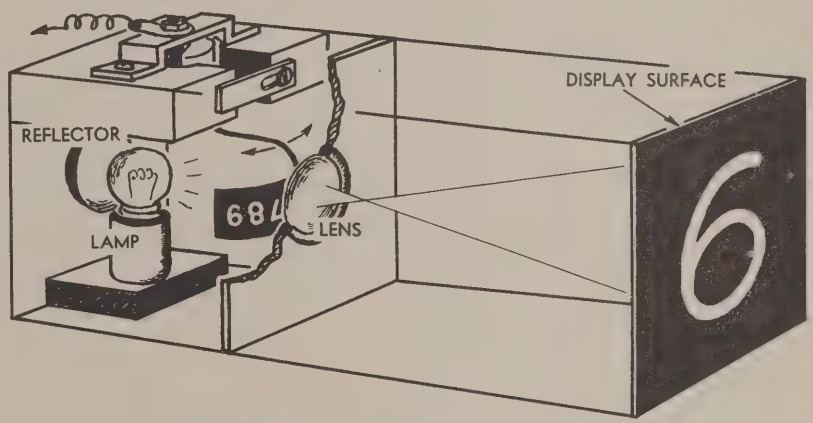


Fig. 10 **MOVING-COIL NUMERICAL INDICATOR**

Finally, the adjustable return voltage is used to adjust the "0" indication by nulling out any residual current which may otherwise be present when the bi-stable stages are all reset. With most practical bi-stable stages there tends to be such a residual current because the output voltage is rarely absolutely zero in the reset state. Usually it is of the order of 500mV, representing the voltage drop across a saturated transistor or other semiconductor device.

SELECTED BIBLIOGRAPHY

DEAN, K. J., **Digital Instruments,** 1965. Chapman and Hall Ltd., London.

"Decade Indicator Operates Off Transistors," in **Electronics**, 3rd November, 1961, p. 60.

GREENBERG, I., "Electroluminescent Display and Logic Devices," in Weber, S., **Optoelectronic Devices and Circuits,** 1964. McGraw-Hill Book Co., Inc., New York.

Chapter 7

BCD - Decimal DECODING

Encoding and decoding—the concept of "sufficient definition" —equivalent functions—choice of logic—sequential or "tree" decoders—parallel decoders—representing a parallel BCD decoder using a matrix—bi-quinary decoding—a decoder for the twisted-ring decade counter.

As we have noted in earlier chapters digital circuitry often deals with electrical representations of numerical information. And in order that the information may be processed with a high degree of reliability and convenience, it is usually handled not in the decimal notation with which human beings are most familiar, but either in **binary** (base-two) notation or in one of the many special notation **codes** developed for this purpose.

In view of the differing requirements of human beings and digital circuitry, and in order that both may operate efficiently, it becomes important to allow each to handle information in the form for which they are best suited, while at the same time providing for effective man-machine communication. An essential part of digital equipment is thus the information **transformation** facilities at both "input" and "output."

In the last chapter, it may be recalled, we looked at the "readout" devices used to produce a visible display from the numerical output of the equipment. Let us now turn to another equally important aspect of transformation: the circuitry used to translate digital representations from one numerical notation system to another.

A suitable name for this aspect of transformation is "code translation." For convenience it is usually conceived as consisting of two complementary operations: **encoding**, wherein the numerical information is translated from decimal notation into a code, and **decoding**—the reverse operation of translating from a code back into decimal notation.

Encoding and decoding being complementary operations, the circuitry involved in the one is usually a functional if not an electrical dual of the other. This being the case, it is not proposed in the present article to deal with both separately. As decoding circuitry is encountered somewhat more frequently than that employed for encoding, it will be the former which will be used mainly as the basis for discussion.

To begin, then. As already stated, decoding is the operation of deriving electrical representations of numerical information in decimal notation from those either in pure binary notation or in one of the many codes. Naturally enough, the aims in decoding are to perform this task both accurately and reliably while using a minimum of circuitry and components. Reference to the diagram of figure 1 and its associated truth table should help in understanding how these aims are achieved.

The diagram may be recognised as the logic diagram of a ripple-carry decade counting register using J-K flip-flops and counting in the so-called 2421 BCD code. The first four data columns of the truth table are by way of recapitulation and show the value states of the flip-flops for each of the ten decimal digits. Note that the lateral order of the flip-flops is transposed in the table to follow the usual convention wherein the bits are arranged from right to left in order of significance or "weighting."

As noted in an earlier chapter the essential point in both counting and other digital circuitry is that there is a unique combination of bits used to represent each decimal digit. Thus inspection of the truth table of figure 1 will show that for every decimal digit there is a unique combination of flip-flop value states.

A moment's thought should show that we can thus derive signals representing the decimal digits from such a register or similar digital circuit, by employing logic circuitry to sense whichever of these unique value state combinations is present at any given time and to deliver an appropriate output. This is the general principle used in decoding.

It might have been assumed from the foregoing that the logical circuitry employed in decoding would have to be arranged to sense the state of **all** the elements in a register or circuit. This is certainly true in the case of pure-binary decoding; however, with many of the BCD and other codes it is often the case that a majority of the decimal digits may be defined adequately by sensing only **some** of the elements. Techniques of logic minimalisation are thus employed to determine the **sufficient definition** for each decimal digit in the code concerned.

For example, in the 2421 BCD code used in figure 1 there is no need to sense the value state of FFD for each of the decimal digits 0—5, because in each case the combination provided by the three remaining elements is sufficient to define the combination uniquely among the ten possible combinations. To define decimal 4, for example, we need only stipulate that FFC must be set (Y=1) and FFB and FFA both reset (Y=0); there is no other decimal digit for which this smaller combination occurs, so it becomes unnecessary to include the value state of FFD in the definition.

For decimals 8 and 9 in the same code only the value states of FFD and

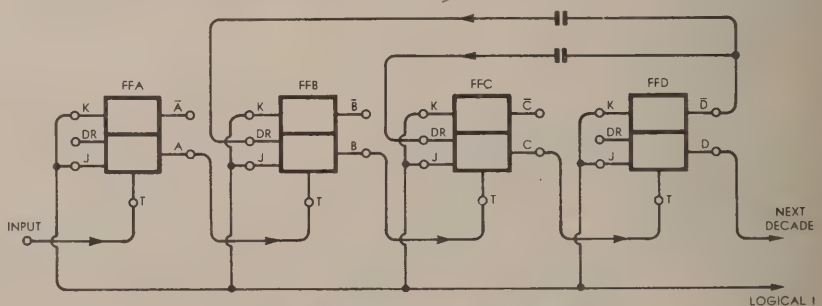


Fig. 1 2421 BCD RIPPLE CARRY REGISTER

DECIMAL	FFD (2)	FFC (4)	FFB (2)	FFA (1)	SUFFICIENT DEFINITION	EQUIVALENT FUNCTIONS
0	0	0	0	0	–000	$\bar{C}.\bar{B}.\bar{A}$, $\overline{C+B+A}$
1	0	0	0	1	–001	$\bar{C}.\bar{B}.A$, $\overline{C+B+\bar{A}}$
2	0	0	1	0	–010	$\bar{C}.B.\bar{A}$, $\overline{C+\bar{B}+A}$
3	0	0	1	1	–011	$\bar{C}.B.A$, $\overline{C+\bar{B}+\bar{A}}$
4	0	1	0	0	–100	$C.\bar{B}.\bar{A}$, $\overline{\bar{C}+B+A}$
5	0	1	0	1	–101	$C.\bar{B}.A$, $\overline{\bar{C}+B+\bar{A}}$
6	0	1	1	0	0110	$\bar{D}.C.B.\bar{A}$, $\overline{D+\bar{C}+\bar{B}+A}$
7	0	1	1	1	0111	$\bar{D}.C.B.A$, $\overline{D+\bar{C}+\bar{B}+\bar{A}}$
8	1	1	1	0	1--0	$D.\bar{A}$, $\overline{\bar{D}+A}$
9	1	1	1	1	1--1	$D.A$, $\overline{\bar{D}+\bar{A}}$

FFA need be defined, as these states are sufficient to differentiate the full combinations concerned from all others. However, in contrast, decimals 6 and 7 can only be defined adequately by using all four of the value states. These results are summarised in the truth table in the column headed "sufficient definition."

Note that whereas sensing the full combinations for each decimal digit would involve the logic circuitry in sensing a total of 40 value states, in fact it is only necessary to sense 30 of these —an average of only three value states per digit and a saving of 25 per cent in the overall operations required from the logic circuitry. Similar savings in circuit complexity and cost are attainable with many other numerical codes. Minimalisation of digit combinations to arrive at their sufficient definitions is thus time and effort well spent.

By its very nature, decoding circuitry is concerned with situations involving logical conjunction: each decimal output must be energised only when there occurs the particular combination of code inputs to which it corresponds. In short, decoding is basically a logical AND function.

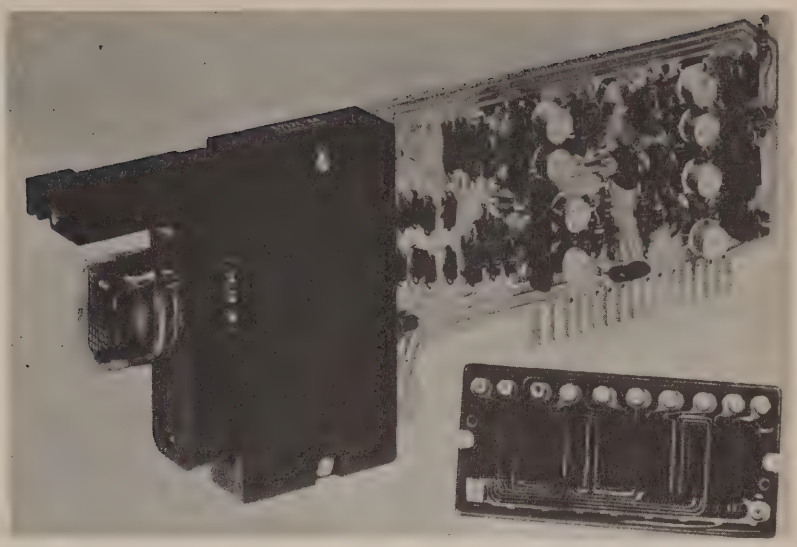
While this may seem to suggest that decoding logic circuitry must use AND elements rather than OR, NAND or NOR elements, this is not the case. As we saw in earlier articles, any logical function may be synthesised using either NAND or NOR elements solely, while the same applies to combinations of AND and NOT or OR and NOT elements. Not only this but most practical logic elements may be arranged to perform any of the logical functions simply by interpreting their behaviour from a suitable logic polarity convention.

From this it should be fairly apparent that there exist a number of equivalent logic functions corresponding to the sufficient definition of each decimal digit. To illustrate this the final data column in the truth table of figure 1 shows both AND and NOR functions equivalent to the sufficient definition concerned from a decoding viewpoint.

The type of logic employed in a decoding circuit is but one of a number of somewhat interdependent factors which are varied by the designer to arrive at the desired performance and reliability with minimum cost.

A second of these factors is the numerical code for which the decoder is being designed. Some codes lend themselves to more economical decoding using AND elements, others to more economical decoding with, say, NOR elements, and so on.

A third factor is the actual logic configuration used. With each code there is usually a particular configuration of one of the types of logic element which will require a minimum of elements or wiring, or both.



This late-model decade counting module by Hewlett-Packard uses a neon lamp-photoconductor "tree" decoder. The photoconductor assembly has been removed to show its construction. (Courtesy Hewlett-Packard Australia Pty. Ltd.)

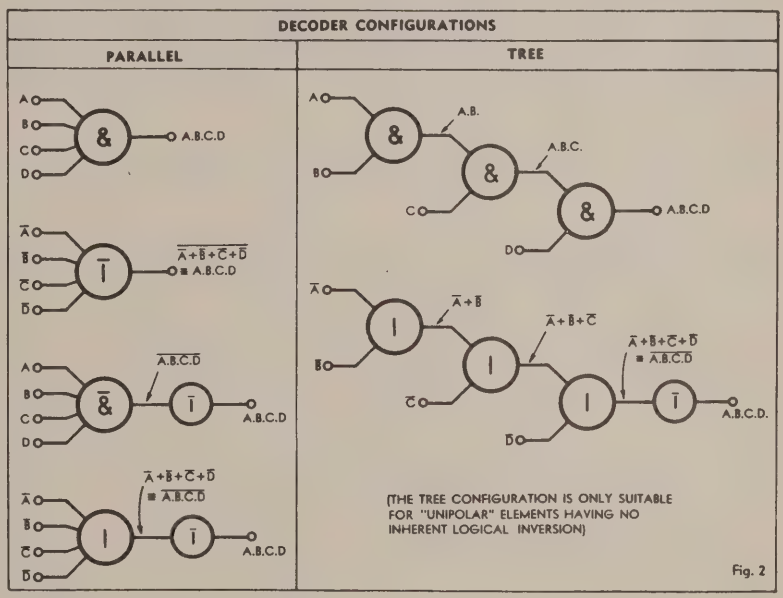


Fig. 2

A fourth closely related factor is the actual device used for performing the logic. The type of device used is often a function of the speed required and cost considerations, while itself determining to a large degree the configuration and type of logic to be used.

The discussion and examples which follow should help the reader in understanding the way in which these factors are inter-related.

There are two general logic configurations available for decoding. These are the **parallel** configuration and the sequential or **tree** configuration. Decoding may be performed using either of these configurations or a combination of both.

The two configurations are illustrated by the diagrams of figure 2, which compare the gating required in each case for a hypothetical decimal digit having a sufficient definition equivalent to A.B.C.D. The alternative diagrams in each column show the way in which various logical elements may be used in the same configuration.

Note that while there are four basic logic types which may be used for the parallel configuration, only two are suitable for the tree configuration: AND and OR. Because of the "distributed" logic performed by the tree configuration the NAND and NOR elements are not suitable, as their inherent inversion results in overall disjunction.

The only way that such elements may be used is by either mixing elements or by combining both parallel and tree configurations — or both. These are further possibilities open to the designer seeking the most economical design.

It should also be noted that the use of some element-configuration combinations requires complementary inputs. While in most decoding situations these are already available from the flip-flops or other source elements, this may not always occur. In such situations it is usually easier to employ a more suitable element-configuration combination than to add the inverters otherwise required.

Perhaps it is worth pointing out at

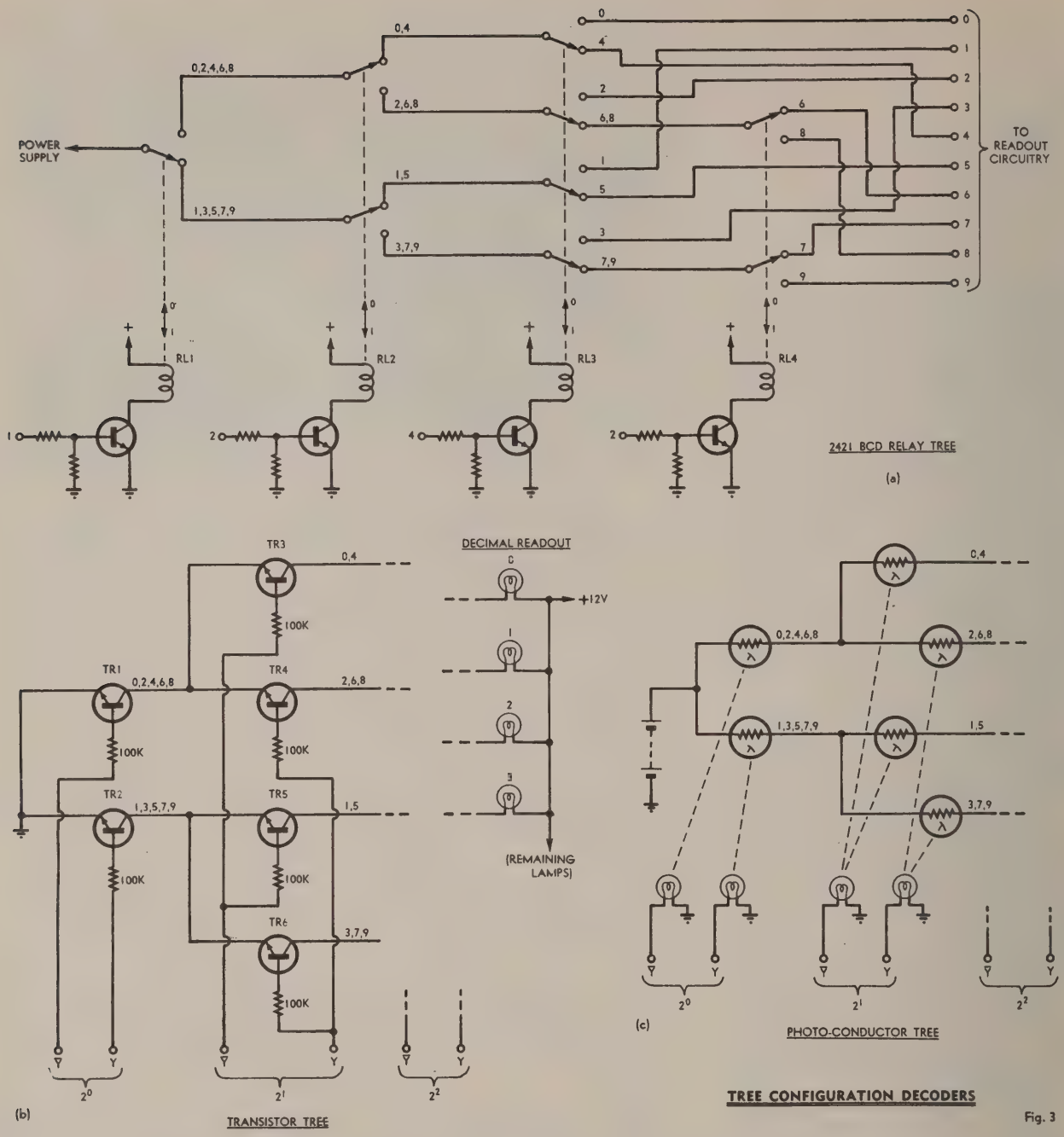


Fig. 3

this stage that most decoders require complementary inputs for at least some of the decimal output digits. Reference to the equivalent functions shown in figure 1 shows the number required in decoding 2421 BCD.

Being a logical operation, decoding may be performed by a variety of different device and circuit elements. Many of the devices and circuits discussed in the second article of this series are thus quite suitable — for example, junction diodes, neon tubes, four-layer diodes, thyristors and transistors. In addition it is often possible to use relatively low speed devices such as relays and photoconductive resistors, as in many decoding applications extremely rapid response is not essential.

The diagrams of figure 3 show three examples of practical decoders which use the tree configuration.

Diagram (a) shows a decoder using relays, which were probably the first devices used in this configuration. Although relatively slow in operation, relays are well suited to the tree configuration as a single relay can operate many sets of changeover contacts.

Each relay changeover contact set is logical equivalent to a pair of alternatively "open" AND gates; the moving contact represents the common input of the two gates, while the two fixed contacts represent the output of the gates. When the relay is not energised, one gate of each "pair" is open and the other closed; energising the relay effectively reverses the situation.

Thus for the 2421 BCD decoder shown, relays RL1, RL2, RL3 and RL4 are effectively controlling two, four, eight and four AND gates respectively. Which decimal output line is connected to the power supply fairly obviously depends upon the combination of signals applied to the relay drivers, energising the relays and opening the appropriate combination of gates.

As shown RL1 and RL3 are energised, effectively opening the three AND gates in the line to the "decimal 5" output terminal. Reference back to figure 1 will verify that 5 is the appropriate decimal digit for this relay energising pattern, which corresponds to 0101 in the code concerned.

The number coding adjacent to each line in the decoder should help in visualising how the decoder operates. Note that the RL1 contact "gates" separate odd numbers from even, and that each set of contacts which follow progressively subdivides the possibilities. Only four inputs are required by the relay tree decoder, derived from the Y outputs of the flip-flops or other elements of the circuit supplying the four bits of code information.

The diagram of figure 3(b) shows part of a decoder using transistors in the tree configuration. The operation is very similar to that of the relay decoder, although the fact that one transistor is required for each AND function results

in two transistors being required to functionally replace each set of relay changeover contacts.

Each bit of the coded input signal provides base bias to saturate appropriate transistors in the tree. Thus the first bit saturates either TR1 or TR2, depending upon its value; the second bit similarly saturates either TR3 and TR5 or TR4 and TR6, and so on. For each coded input number there will thus be an appropriate combination of saturated transistors giving a current path through the tree.

It may be seen that transistor tree decoder **grounds** the selected output connection rather than connecting it to a power supply as did the relay decoder; it might thus be called a "negative logic" decoder. This change is dictated by the gating bias requirements of the transistor elements.

A complication arising in connection with the gating bias is that later transistors in the tree receive a progressively diminished gating input voltage as a result of the voltage drop across earlier transistors. The reason for this is that each transistor acts as an emitter series impedance for those which succeed it.

Because of this complication the transistor tree decoder is generally unsuitable for decoding information in the form of large (i.e., many-bit) numbers. For four-bit BCD decade decoding it is usually quite satisfactory, although it tends to be somewhat costly compared with alternative systems. In practice pure transistor tree decoders are rarely used; more usually the tree approach is used to simplify a basically parallel configuration.

Figure 3(c) shows part of a third type of tree decoder which is currently gaining in popularity by virtue of its rather unique combination of economy, simplicity and reliability. It is also very suitable for many of the fabrication techniques used in the manufacture of microcircuits.

As may be seen the circuit is rather similar to that of the transistor decoder. The essential difference is that the AND elements are here photo-conductors. One input to each photo-conductor is the applied voltage derived from the fixed power supply; the other is light received from the associated lamp.

In darkness, each photo-conductor element possesses a high resistance and passes negligible current; however when illuminated its resistance drops markedly and it becomes capable of passing appreciable current. Because of the lamp-element grouping, each coded input number results in the illumination of the correct combination of elements to provide a low resistance path to the appropriate output terminal.

Although the lamps shown in the diagram are of the incandescent variety, almost any type of lamp is suitable; the choice of lamp type is dictated mainly by the type of drive circuitry available. Where sufficient drive voltage is available miniature neon lamps are often the most logical choice. Newer alternatives which are potentially better suited to microminiaturisation techniques are electroluminescent panels and light-emitting semiconductor junction diodes.

Let us now consider practical aspects of the parallel decoder configuration. As noted in connection with the diagrams of figure 2, this configuration is somewhat more flexible than the tree type in that it may be constructed using all

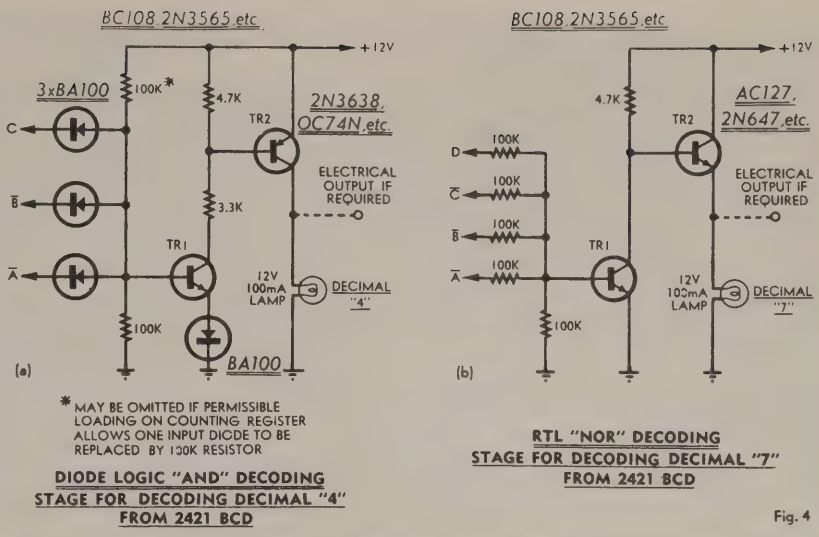
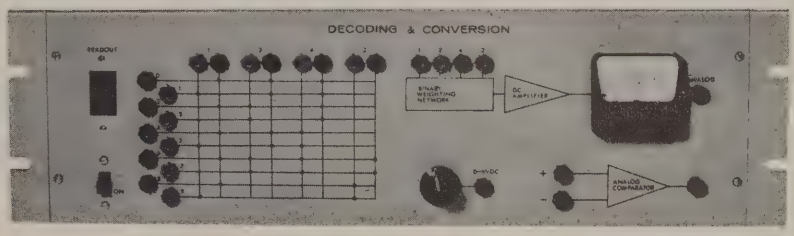


Fig. 4



Shown here is the third panel from the author's digital demonstrator unit, which is described in chapter 12. The left-hand portion is a BCD-decimal decoder using circuitry as in (a) above.

types of combinational logic element — NAND and NOR elements are suitable as well as those for AND and OR.

While this means that there is an even larger range of potentially suitable logic devices and circuit elements than with the tree configuration, there are only two general circuits in common use. One uses diode gates to perform straightforward AND logic; the other uses resistor-transistor (RTL) gates to perform NOR logic on the complementary inputs. A sample stage from simple decoders of each type is shown in figure 4.

Diagram (a) shows the circuit for a complete diode logic AND stage to decode decimal "4" from 2421 BCD. It consists of the actual diode gate together with a two-transistor driver amplifier feeding a 12V 100mA readout lamp.

As shown the gate has three diode inputs, which connect to the code terms constituting the equivalent AND function for decimal "4" in 2421 BCD. However, if it is permissible to load the code sources slightly, one of the diodes may be replaced by the 100K resistor shown connecting to +12V. This leaves the logical operation unchanged but saves the cost of one diode.

The non-inverting driver amplifier which follows the gate was discussed in the preceding article of this series,

GLOSSARY OF IMPORTANT TERMS

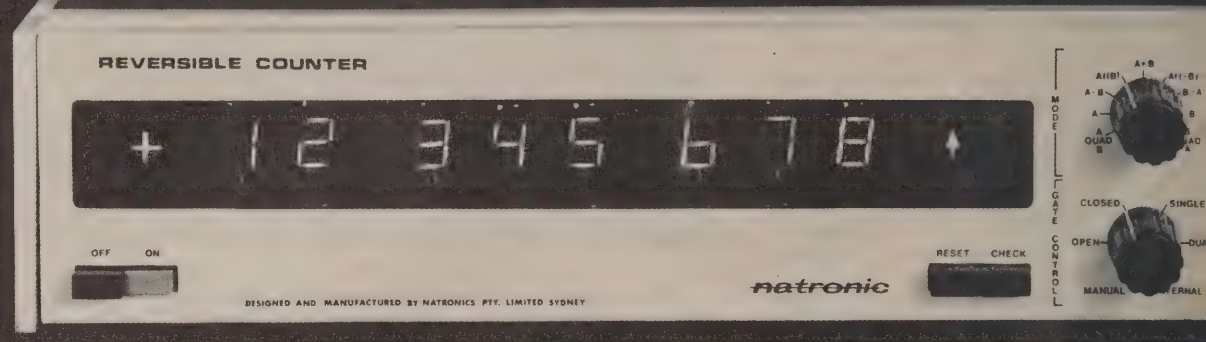
Code translation: Translation of information from one (usually numerical) notation system into another. Consists of the complementary operations **encoding** and **decoding**, the former concerned with translation of information into a code and the latter concerned with translation from a code.

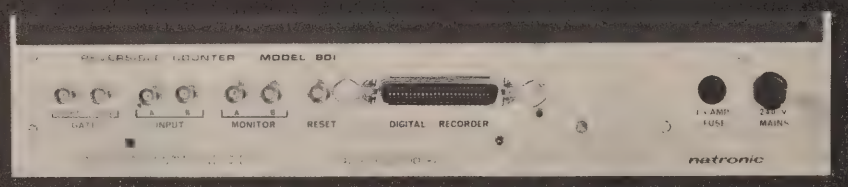
Parallel Decoding: A method of decoding using logic configurations which sense each meaningful code combination by effectively performing a simultaneous AND operation.

Sequential or Tree Decoding: A method of decoding which uses a distributed sequence of AND operations to sense the meaningful code combinations.

Sufficient Definition: As a consequence of the redundancy inherent in many digital codes it is not necessary during decoding to sense all code bits. Each combination need be detected by sensing only those bits whose occurrence is sufficient to define the combination concerned.

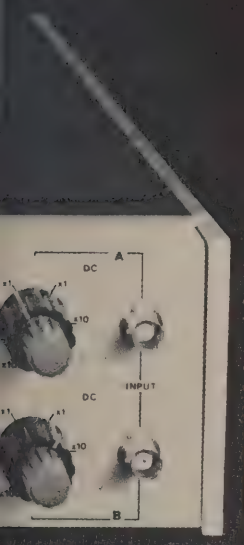
natronics reversible c
negatively a





IRH THE EQUIPMENT DIVISI

unter totalises positively and nd counts through zero from ther direction — D.C. to 2Mhz.



The Natronics Reversible Counter has virtually unlimited application in research and industry. It adds, subtracts, compares rates, compares variables against fixed references, indicates absolute position, integrates noisy signals despite excursions below zero, determines and controls X and Y co-ordinates.

APPLICATIONS INCLUDE . . .

- Numerical control of machine tools
- Laser inferometry
- Tank level indication
- Radiation counting — eliminating background radiation
- Carbon 14 dating
- Astronomy — measurement of light intensity
- Magnetic field strength indication
- Moire fringe techniques
- Comparison of speeds
- Integration of noisy signals (despite excursions above and below zero)
- Determination of X and Y co-ordinates
- Meteorological measurements
- Readout from phase sensitive incremental transducers
- Deviation monitoring

OPERATING MODES

A quad B: Totalises A as a function of the phase of B.
Totalises A positively if B leads A.
Totalises B negatively if B lags A.

B quad A: As above but with count direction reversed.

A: Totalises A only.

B: Totalises B only.

A + B: Totalises A and B—circuitry prevents count loss in the event of co-inciding pulses.

A − B: Totalises A minus B—circuitry prevents count loss in the event of co-inciding pulses.

B − A: Totalises B minus A—circuitry prevents count loss in the event of co-inciding pulses.

Af(B): Totalises A as a function of B from D.C. to 2Mhz. If B is positive, A is totalised positively. If B is negative, A is totalised negatively.

Af(− B): Totalises A as a function of − B from D.C. to 2Mhz. If B is positive, A is totalised negatively. If B is negative, A is totalised positively.

GENERAL

Range: D.C. to 2Mhz on both input channels.

Indication: 8 "Digivac" electro-luminescent long life tubes, + and − indicators, ↑↓ gate overflow indicators.

Reset: "Manually" by front panel mounted push button.
"Remotely" by minimum 2.4 volt pulse via rear mounted BNC connector.

Input protection: Input is FET, protected by resistor and diode networks.

Print level: Compatible with DTL or TTL logic.

Temperature range: 10°C to 50°C.

RFI: Fully shielded inputs, instrument unaffected by electromagnetic radiation levels up to 200mV/metre.

Dimensions: Width 19", height 3½", depth 14".

INPUT CHANNELS

D.C. coupled: 0 − 2Mhz.

A.C. coupled: 10hz − 2Mhz.

Input impedance: 1 Megohm shunted by 30pf.

Maximum input: A.C. coupled 600 volt peak.
D.C. coupled 500 volt peak.

Sensitivity: 0.1 volt r.m.s. sine wave, 1 volt pulse, 0.2 micro-second minimum width, level sensitivity x1, x10, x100.

Trigger level: − 1.0 volt to + 1.0 volt x1, x10, x100, independently adjustable for each input channel.

GATE CONTROL

Manual: Controlled by front panel mounted function switch for "open" and "closed" modes.

External single: BNC connector provided on rear panel for "start" input.

External dual: Individual BNC connectors on rear for "stop" and "start" inputs.

Input impedance: Greater than 10K.

Trigger level: Minimum level 2.4 volt pulse, maximum input 100 volts.

Gate logic: Open—"1"—positive input 2.4 volts; close—"0"—positive input less than 0.4 volt.

PRINTER OUTPUT

Output format: Four line, 1-2-4-8 BCD.

"0" State: Approx. + 0.4 volt.

Single line output: + level approx. 0.4 volt; − level approx. 2.4 volt.

Overflow: Single line output—"off" level approx. 2.4 volt; "on" level approx. 0.4 volt.

Strobe inhibit: Prevents change during print-out, operative when signal exceeds 2.4 volts.

*** Ex stock delivery**

The Natronics Reversible Counter is designed and manufactured in Australia by Natronics Pty. Limited — the equipment division of IRH Industries Ltd.

NATRONICS PTY. LIMITED
THE CRESCENT, KINGSGROVE, N.S.W. 2208. TEL. 50 0111

OF IRH INDUSTRIES LTD.

WA 0243

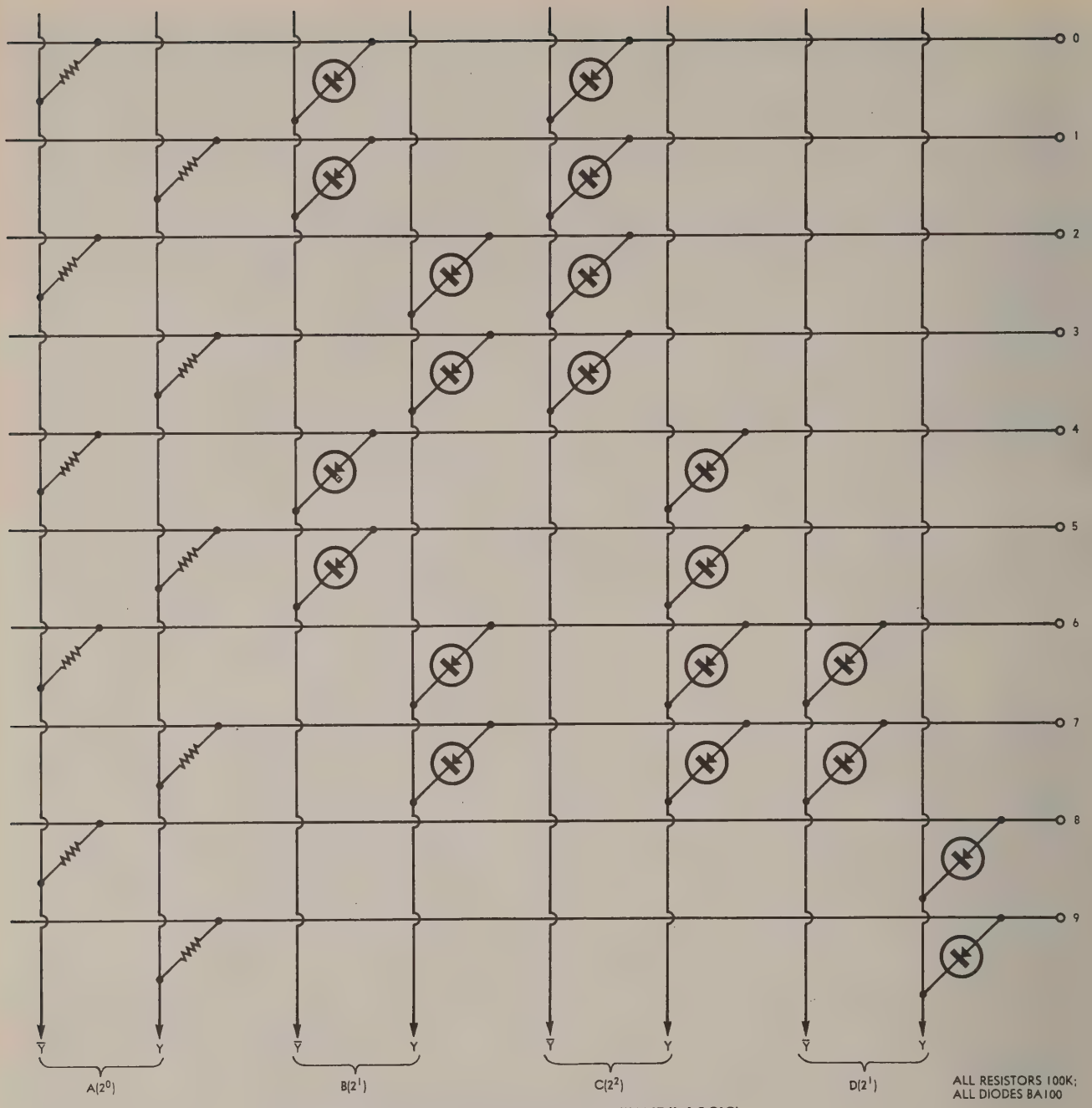


Fig. 5 **2421 BCD DECIMAL DECODING MATRIX ("AND" LOGIC)**

and should need little further discussion here. The silicon diode in series with the emitter of transistor T1 provides the transistor with hold-off bias to ensure that it does not conduct unless all inputs are at logical "1" (=+9V). It is necessary because the code inputs do not in practice fall to 0V in the "0" state, but rather to a level representing the voltage drop across a saturated transistor.

The diagram of figure 4(b) shows for comparison the circuit for an RTL decoder stage. Here the stage is designed to decode 2421 BCD decimal "7," and has accordingly four inputs corresponding to the four terms of the NOR equivalent function (see figure 1).

The circuit may be recognised as differing only slightly from the NOR gate circuit given in the second article of this series. The input resistors perform an OR operation, transistor T1 performs inversion and transistor T2 provides current amplification to drive the indicator lamp.

In general, NOR parallel decoders are more attractive than AND decoders from a cost viewpoint, as they can use low-cost resistors in place of semi-conductor diodes. However where high-speed operation is required this advantage tends to disappear as the resistors must be shunted with compensating capacitors if the RTL circuit is to operate effectively.

Even at lower operating speeds the diode AND logic may prove more attractive, in that it lends itself somewhat more conveniently to a mixed parallel-tree configuration. Using such a mixed configuration with AND logic it is often possible to arrive at a more economical design than if a pure-parallel RTL configuration is used.

Unfortunately space limitations will not permit more than a brief reference to mixed configuration decoders at this point. However, an example of this type of decoder will be discussed shortly in connection with bi-quinary decoding.

Before this is done mention should be made of the common practice of conceiving and representing parallel BCD decoders using matrices.

In mathematics, when a relatively large number of terms have to be handled ed or processed or allowed to interact in a similar fashion, it often proves convenient to group them into a square or rectangular grid array which is called a matrix. By observing various conventions it is usually possible to treat the numbers far more easily and effectively as a matrix than individually.

The advantages to be gained from grouping things together as matrices are not peculiar to mathematics, however, and many other fields have accordingly "borrowed" the concept. In electronics it is finding increasing use as circuitry becomes more complex and repetitive; this particularly true in digital electronics.

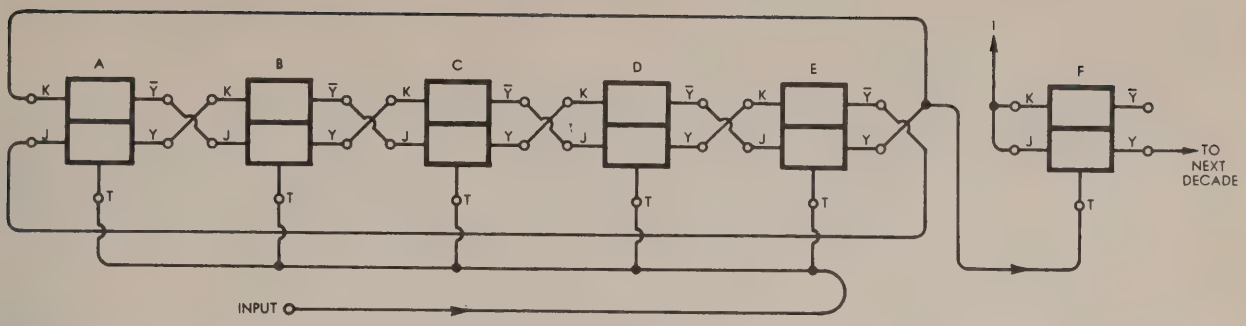


Fig. 6

BI-QUINARY DECADE COUNTER

A good example of the application of matrices in this field is that of the ferrite core memory, where large numbers of tiny ferrite annuli ("doughnuts") are used to store binary or BCD information. It proves most convenient both from a fabrication and from a functional viewpoint to group large numbers of the cores together as matrices—not only are they grouped together compactly in a single frame, but it is relatively easily arranged that they share much of the required input and output processing circuitry.

It is similarly advantageous to apply the matrix concept to parallel type BCD decoders, grouping the decoder stages for all the output digits together as shown in figure 5.

The diagram shown is that for the decoder panel shown in the photograph, forming part of the author's "digital demonstrator" unit. As may be seen it is simply the combination of ten AND stages of the type shown in figure 4(a). The resistor-diode substitution has been incorporated for economy reasons, with the resistors used in each case for the "A" or least-significant-bit (LSB) input.

Note that the matrix is used to group together only the actual decoding gates. The readout driver amplifiers thus become identical and are simply connected to each output terminal.

To conclude this discussion of decoding it may be worthwhile to give examples of decoders for non-BCD decimal codes. The first of these to be discussed is that for the bi-quinary (5 x 2) code.

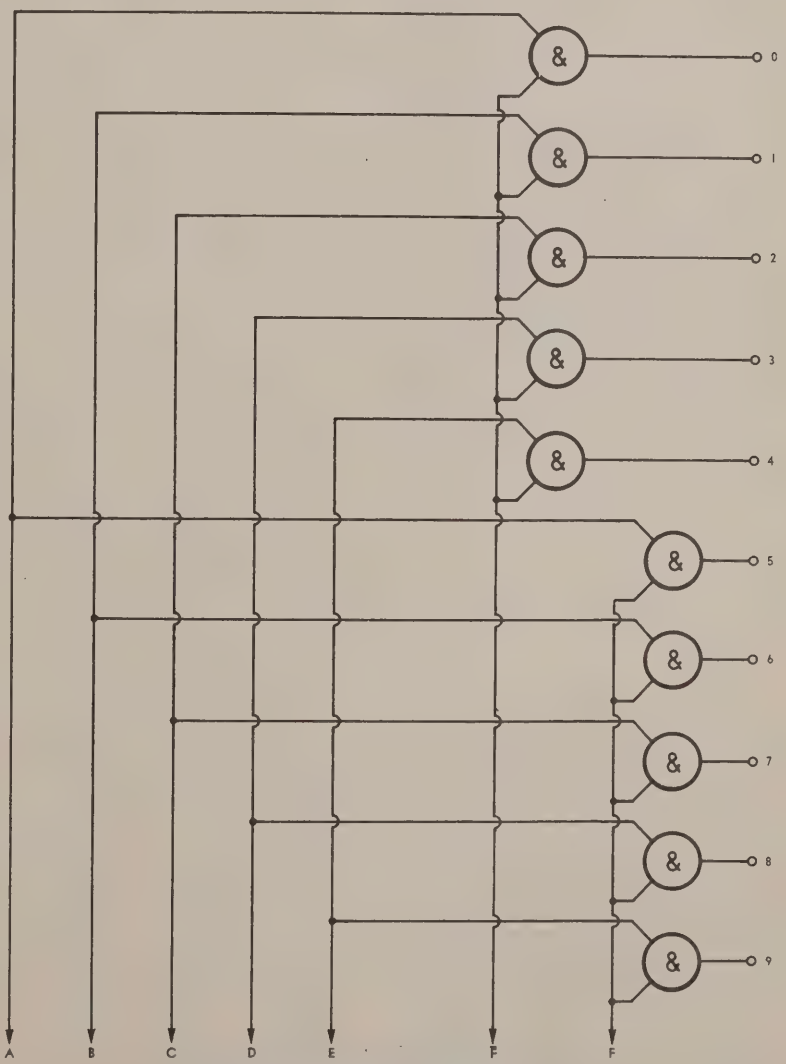
The diagram of figure 6 shows the basic logic of a bi-quinary counting decade using J-K flip-flop elements. Six elements are used in all, with five wired together as a normal ring counter and the remaining element following as a divide-by-two complementing stage.

As the first set of truth table columns shows, this combination of elements gives ten unique value-state combinations which correspond to the bi-quinary decimal code mentioned in earlier articles. Each combination consists of six value-states.

To decode the bi-quinary counter it is not necessary to sense all six value states of each combination. Close inspection reveals that only two value states are required to provide a sufficient definition for each decimal, as shown in the seventh data column.

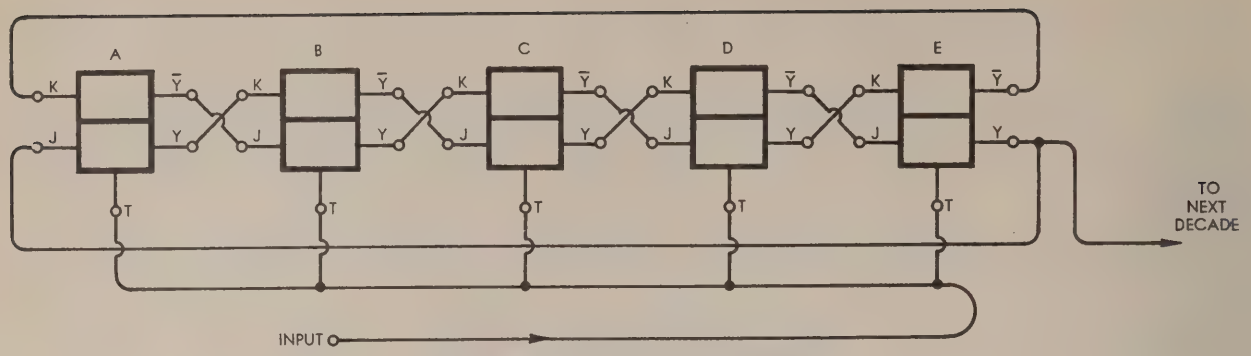
As before there are various logic functions equivalent to the definition, corresponding to the various possible decoding logics. The AND and NOR functions are shown in the final column of the truth table.

DECIMAL	A	B	C	D	E	F	SUFFICIENT DEFINITION	EQUIVALENT FUNCTIONS
0	1	0	0	0	0	0	1----0	A.$\bar{F}$, $\overline{\bar{A}+F}$
1	0	1	0	0	0	0	-1---0	B.$\bar{F}$, $\overline{\bar{B}+F}$
2	0	0	1	0	0	0	--1--0	C.$\bar{F}$, $\overline{\bar{C}+F}$
3	0	0	0	1	0	0	---1-0	D.$\bar{F}$, $\overline{\bar{D}+F}$
4	0	0	0	0	1	0	----10	E.$\bar{F}$, $\overline{\bar{E}+F}$
5	1	0	0	0	0	1	1----1	A.F, $\overline{\bar{A}+\bar{F}}$
6	0	1	0	0	0	1	-1---1	B.F, $\overline{\bar{B}+\bar{F}}$
7	0	0	1	0	0	1	--1--1	C.F, $\overline{\bar{C}+\bar{F}}$
8	0	0	0	1	0	1	---1-1	D.F, $\overline{\bar{D}+\bar{F}}$
9	0	0	0	0	1	1	----11	E.F, $\overline{\bar{E}+\bar{F}}$



BI-QUINARY DECODER ("AND" LOGIC)

Fig. 7



TWISTED-RING DECADE COUNTER

Fig. 8

Shown in figure 7 is the logic diagram for a bi-quinary decoder using the AND equivalent functions. Although a very simple decoder it may be seen to possess some of the features of both the tree and the parallel configurations, and might thus be regarded as of the "mixed" type.

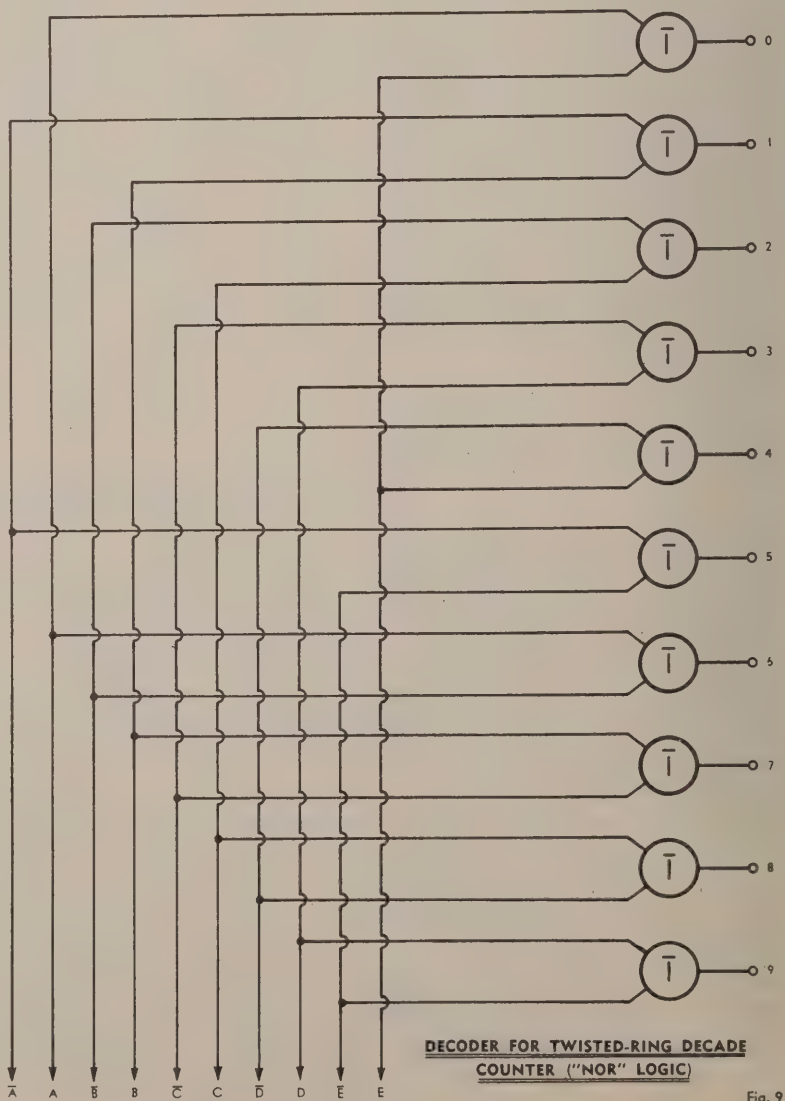
Finally let us look briefly at a second example of non-BCD decoders: a decoder designed for the twisted-ring decade counter. The logic diagram and truth table for such a counter are shown in figure 8.

As we saw in an earlier article, the twisted-ring counter uses five bi-stable elements connected to form a ring possessing a single logical inversion. It provides the required ten unique value states with the important feature that only one element changes state upon the arrival of any input pulse. The latter feature makes it eminently suitable for high-speed counting and scaling.

Decoding the twisted-ring counter is as simple as with the bi-quinary counter. As with the latter, only two value states per combination are required to form the sufficient definition of the appropriate decimal digit. The sufficient definitions and corresponding AND and NOR equivalent functions are shown in the truth table.

Figure 9 shows a twisted-ring decoder using NOR logic. Here again while being a very simple decoder it is perhaps most conveniently regarded as of the "mixed" configuration type.

DECIMAL	A	B	C	D	E	SUFFICIENT DEFINITION	EQUIVALENT FUNCTIONS
0	0	0	0	0	0	0 – – – 0	$\overline{A}.\overline{E}$, $\overline{A+E}$
1	1	0	0	0	0	10 – – –	$A.\overline{B}$, $\overline{\overline{A}+B}$
2	1	1	0	0	0	– 10 – –	$B.\overline{C}$, $\overline{\overline{B}+C}$
3	1	1	1	0	0	– – 10 –	$C.\overline{D}$, $\overline{\overline{C}+D}$
4	1	1	1	1	0	– – – 10	$D.\overline{E}$, $\overline{\overline{D}+E}$
5	1	1	1	1	1	1 – – – 1	$A.E$, $\overline{\overline{A}+\overline{E}}$
6	0	1	1	1	1	01 – – –	$\overline{A}.B$, $\overline{A+\overline{B}}$
7	0	0	1	1	1	– 01 – –	$\overline{B}.C$, $\overline{B+\overline{C}}$
8	0	0	0	1	1	– – 01 –	$\overline{C}.D$, $\overline{C+\overline{D}}$
9	0	0	0	0	1	– – – 01	$\overline{D}.E$, $\overline{D+\overline{E}}$



DECODER FOR TWISTED-RING DECADE COUNTER ("NOR" LOGIC)

Fig. 9

SELECTED BIBLIOGRAPHY

DEAN, K. J., **An Introduction to Counting Techniques and Transistor Circuit Logic,** 1964. Chapman and Hall Ltd., London.

GILLIE, A. C., **Binary Arithmetic and Boolean Alegbra,** 1965. McGraw-Hill Book Co., Inc., New York.

HARRISON, B. H., "Photoconductive Matrix Simplifies Counter Display," in Weber, S., **Optoelectronic Devices and Circuits,** 1964. McGraw-Hill Book Co., Inc., New York.

MILLMAN, J., and TAUB, H., **Pulse, Digital and Switching Waveforms,** 1965. McGraw-Hill Book Co., Inc., New York. (Ch's 8, 9, 18.)

Chapter 8

Digital - Analog CONVERSION

Digital-to-analog conversion—the weighted resistor DAC—the ladder network DAC—a simple practical circuit—analog-to-digital conversion—the analog comparator—a simple circuit — the simultaneous ADC — the counter-type ADC — the continuous ADC—other types.

It may be recalled that in the last chapter we looked at the complementary processes of encoding and decoding—two important aspects of a general digital procedure often given the name "information transformation." Let us now turn our attention to a related aspect of the same general procedure, an aspect which arises whenever information must be transformed either from digital to analog form or vice-versa: **digital-analog conversion.**

By way of recapitulation, it should perhaps be noted at this point that information in digital form usually may possess only two values, whereas information in analog form may possess a large or possibly infinite number of significant values.

As one might expect, conversion is similar to code translation in that there are two complementary processes — digital to analog conversion or "DAC," and analog to digital conversion or "ADC."

Digital-to-analog conversion finds its main use wherever information in digital form must be fed to, or used to control, equipment which accepts information most conveniently in analog form. Examples of this are where paper-chart recorders are to be used to plot continuous graphs of digital information and in digital machine control where a digital control signal may be required to drive a motor or solenoid unit.

A secondary application of DAC, but one which is rapidly growing in importance, is the generation of analog signals which must conform with a high order of precision to given mathematical functions. Where the function concerned is a highly complex one and/or where high orders of precision are required, generation of the signal by digital means and subsequent DAC will often give a superior result to more conventional analog generation techniques.

An example of this is where a highly linear voltage ramp or gradient must be generated, particularly over an extended period of time. Using conventional analog techniques, this is quite a difficult task, but using a DAC approach it can be relatively easy. A large-capacity pure binary counter fed with pulses at a suitable rate and itself driving a many-bit DAC will produce an accurate "staircase" voltage or current, the steps of which can be made small enough to be negligible for the application concerned.

Similarly, a DAC approach is sometimes the only feasible way of generating or synthesising a complex non-periodic waveform.

Quite apart from its direct applications, DAC has an indirect application in the complementary process of analog-to-digital conversion. The reason for this is that most ADC's employ a DAC as a feedback element, as will be explained later in this chapter.

Fundamentally, a DAC constructs an analog equivalent of a digital input signal by adding together voltages or currents having magnitudes proportional to the weights of those bits in the digital information which are logically true (=1) at the time concerned. Thus if the digital input is decimal number 5, perhaps encoded in pure binary or a BCD code, the DAC constructs an analog equivalent by adding together four units and one unit of voltage or current.

In practice, most DACs employ current addition as this usually proves somewhat easier to perform than voltage addition. The diagram of figure 1 shows the general principle of operation of a current-adding DAC for a single decade of 2421 BCD code.

As may be seen, it consists of four constant-current generators controlled by the four input BCD bits. The generator controlled by the first or least significant (LSB) bit, when gated, produces a constant current "I"; those for the second, most significant (MSB) and fourth bits produce "2I," "4I" and "2I" respectively, corresponding to the appropriate bit weighting.

The outputs from the four generators are connected together so that they add to produce an output "nI," where n is the decimal number represented by the BCD input. To produce an analog voltage signal from this current signal, it is simply necessary to feed the current through a resistor R, giving an output nE where E=IR.

There are two methods used in practice to perform the general operation shown in figure 1. These are usually known as the "weighted resistor" and "ladder network" methods, and are illustrated in figure 2. Both diagrams show four-bit converters, the first for 2421 BCD and the second for 8421 BCD.

In the weighted resistor DAC each input bit operates a switch connecting one of four precision resistors to a precision voltage supply. The resistors connect to the input of a high-gain DC amplifier, across which is fitted a negative feedback resistor Rf.

If the DC amplifier is assumed to have infinite gain, the effect of the feedback resistor Rf is to give the amplifier a virtual input resistance of zero. Its active input terminal thus remains at earth potential regardless of the current supplied by the input resistors. Under these conditions the current controlled by each of the input switches will be independent of any current flowing through the other switches.

Since the input resistors have values weighted inversely for the BCD code concerned, this means that each switch can provide the amplifier with a constant current whose magnitude is proportional to the weighting of the corresponding BCD bit. If the output of the precision voltage supply is E volts and E/R=I, switch Sa will therefore control a current of I while Sb, Sc and Sd will control currents of 2I, 4I and 2I respectively.

The input current to the DC amplifier is thus nI, where n as before corresponds

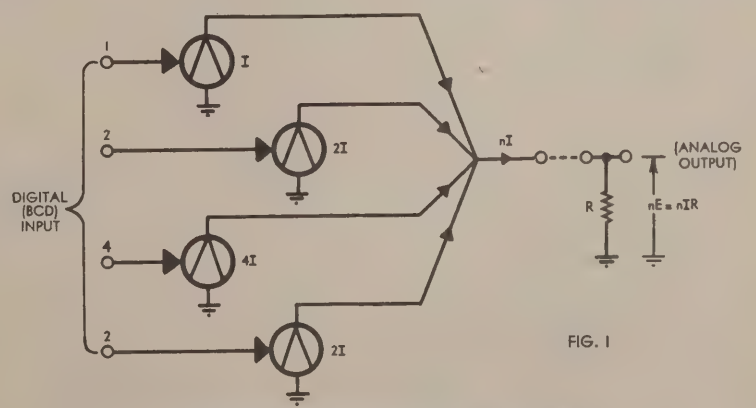


FIG. 1

GENERALISED DIGITAL-TO-ANALOG CONVERTER

(SHOWN FOR 2421 BCD CODE)

An important application of digital-to-analog conversion is where digital information is to be displayed via a cathode-ray tube. The position of the tube electron beam is defined by binary numbers, which are converted by DACs to produce corresponding analog deflection currents.

to the number encoded in the BCD input bits.

Because of the negative feedback resistor Rf, the amplifier will convert this analog current input to an analog voltage output nE', where E' will be equal to IRf. The reason why the amplifier produces a voltage output of nE' is that this output voltage produces a feedback current through Rf which is just sufficient to cancel the input current nI and maintain the amplifier input at effectively zero potential.

As may be seen, the DC amplifier acts as an "operational" amplifier, connected to the precision reference voltage E with a voltage gain equal to the ratio of feedback resistor Rf to the effective input resistor R/n. The analog output voltage may thus be expressed as

$$nE' = n.\frac{E.Rf}{R} \quad \ldots (1)$$

In other words, the output is directly proportional to the digital input number **n.** In the foregoing discussion, it was assumed that the voltage gain of the DC amplifier was infinite; whereas, in fact, no practical amplifier has infinite gain. This means that, in practice, the output voltage of a simple weighted resistor DAC tends to be non-linear, giving errors in the analog output corresponding to each input number.

The errors are generally small enough to be negligible providing the amplifier gain is higher than about 10,000. However, for applications requiring extreme accuracy, the basic circuit configuration can be modified in order to render the conversion linearity independent of amplifier gain.

The modification required for this is shown in dashed lines. As may be seen, it simply involves earthing the ends of the unused input resistors, to provide across the amplifier input an effective shunt resistance of R/(9—n).

The shunt resistance, in conjunction with the effective input resistance R/n forms a voltage divider across the precision supply. The divider action is such that the amplifier receives input currents which are directly proportional to the BCD input bit weightings, despite the non-zero value of the amplifier input voltages.

The analog output voltage nE' thus becomes exactly proportional to the digital input number **n**, as in the ideal case. It may be expressed as

$$nE' = n.\frac{A.E.Rf}{9Rf+R(A+1)} \quad \ldots (2)$$

The second type of DAC method is the "ladder network" method, as illustrated in the lower diagram of figure 2. Here the current contributed to the DC amplifier by each switch is determined not just by the specific value of its series resistor, but also by the position of the resistor in a network arranged as a ladder-type attenuator.

The ladder is usually arranged so that there is a constant $\times 2$ (or 6dB) division between successive steps. Actually it is rather difficult to use a ladder giving unequal steps, or even equal steps of other than 6dB — a fact which makes the ladder-type DAC suitable only for pure binary or 8421 BCD conversion.

In the diagram, switch Sd controls the largest input current contribution, being nearest to the amplifier input; it therefore corresponds to the MSB input. In contrast switch Sa controls the smallest contribution, and therefore corresponds to the LSB input.

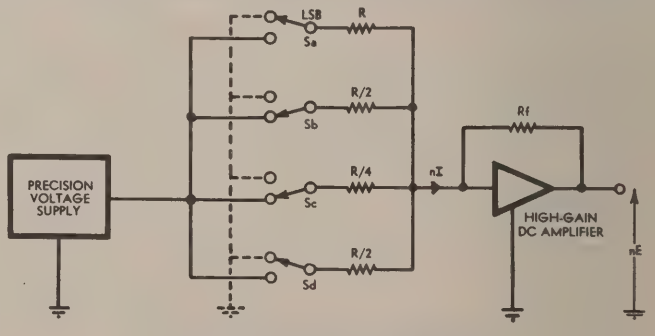
Using the same symbols as before, the analog output voltage of the ladder network DAC shown can be expressed as

$$nE' = n.\frac{A.E.Rf}{16(Rf+R(A+1))} \quad \ldots (3)$$

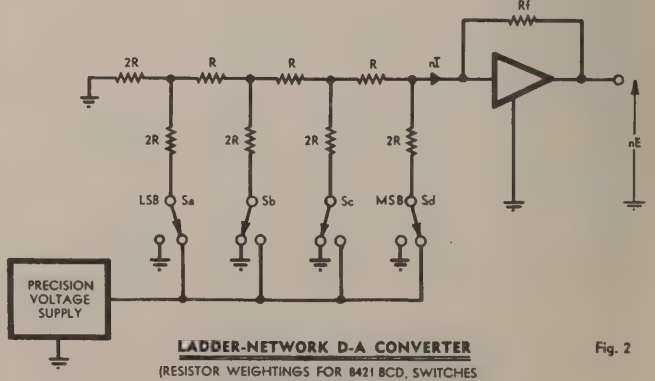
As may be seen, the output is again directly proportional to the digital input number **n**, and conversion is linear for any value of amplifier gain **A.**

In comparison with the weighted resistor DAC the ladder network configuration tends to have a more uniform current and dissipation distribution; all switches and resistors can thus be designed for approximately the same currents. Also, the weighted resistor DAC uses five different precision resistor values per decade of four bits, whereas the ladder network DAC uses only three different values; this can in some cases

The two methods most commonly used to perform DAC. While the second method is somewhat better suited to high speed conversion, it is less flexible in terms of acceptable input coding.



WEIGHTED-RESISTOR D-A CONVERTER
(RESISTOR WEIGHTINGS FOR 2421 BCD, SWITCHES SHOWN IN POSITIONS FOR DECIMAL "6")



LADDER-NETWORK D-A CONVERTER
(RESISTOR WEIGHTINGS FOR 8421 BCD, SWITCHES SHOWN IN POSITIONS FOR DECIMAL "9")

Fig. 2

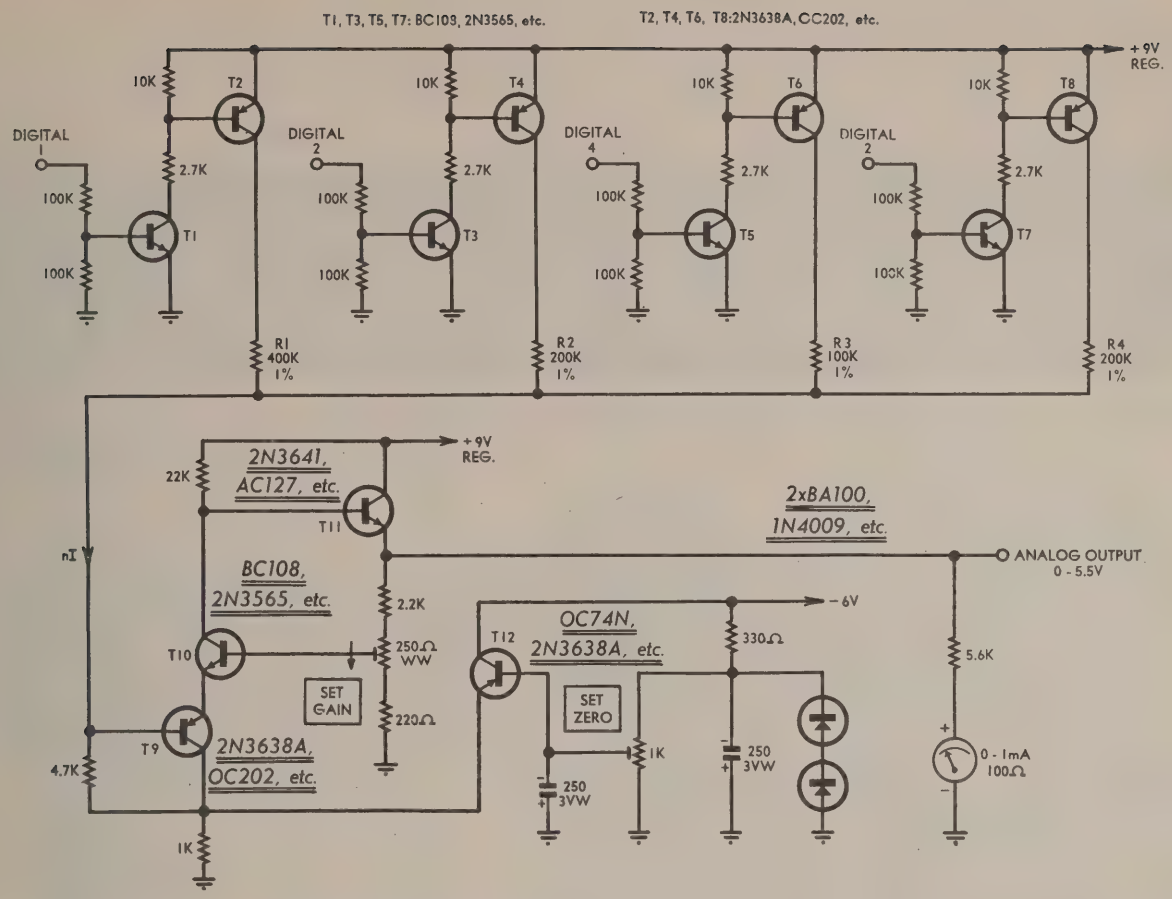


Figure 4: The circuit of a simple DAC for 2421 BCD code.

make the ladder network DAC a more economical proposition.

Although both the DAC examples given in figure 2 are only four-bit converters, intended for conversion of a single decade of BCD code input, both configurations can be used for converters dealing with larger numbers of bits.

To extend the capacity of the weighted resistor configuration, it is simply necessary to add further switches and resistors of appropriate value. However, with the ladder configuration, simple expansion is possible only for pure-binary or similar continuous codes. Multi-decade BCD or similar decimal code conversion must be performed using multiple 4-bit converters.

In both the weighted resistor and ladder network DACs the input switching may be performed in a variety of ways. The main requirement is for switches which are as close as possible to an ideal switch—SPST in the case of the simple weighted resistor DAC, or SPDT in the case of the modified weighted resistor and ladder network DACs.

In slow-speed conversion applications, relays and transistor drivers may be used as in figure 3(a). A driver and relay as shown are required for each input bit, the relay contacts being either SPST or SPDT as required.

Where operating speed and reliability are important, transistor switching is usually employed. Figure 3(b) shows a suitable configuration, again for a single input bit. For SPDT switching, an NPN transistor is added as shown by the dashed lines.

While more reliable and capable of extremely high operating speeds, transistor switching invoives problems arising from the fact that transistor switches do not have zero voltage drop even when fully saturated. The transistor saturation voltages introduce a further source of DAC non-linearity, particularly in the case of the weighted resistor configuration.

Despite this, however, transistors are the most often used switches in both types of converter. By careful design the errors produced by transistor saturation voltages are kept to an acceptably low level.

Figure 4 shows the circuit of a practical DAC of the simple (SPST) weighted resistor type. The circuit uses transistor switching and is designed for 2421 BCD conversion using positive input logic. In fact, it forms the major part of the "Conversion" section of the third panel of the author's digital demonstration unit, the construction of which will be discussed later.

Transistor pairs T1-T2, T3-T4, T5-T6 and T7-T8 are the input driver-switch combinations, in each case turning the second transistor "on" when logical 1(=+9V) is applied to the input terminal. The emitters of the switch transistors connect to the +9V line, which acts as the reference voltage E; the collectors connect to the appropriate high-stability resistors R1, R2, R3, R4.

Transistors T9-12 form a simple DC amplifier. T9 is an input emitter follower, in "inverted" configuration to give low "offset." The latter term describes a range in input quantity values which is treated by a device as if all the values included do not differ from zero—it is

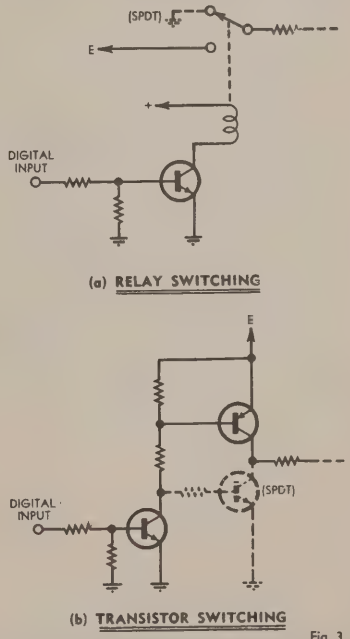


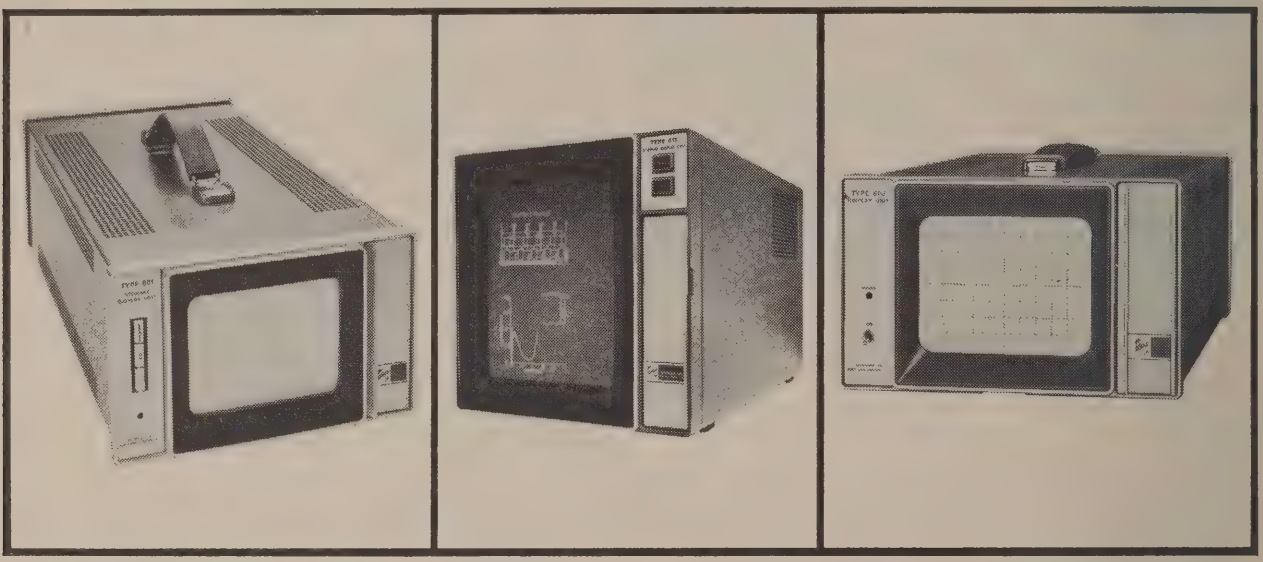
Fig. 3

largely a function of the turn-on voltage of transistor and diode junctions.

T10 is a feedback mixer and voltage amplifier used to stabilise and adjust the gain; T11 is an output emitter follower, while T12 is used to provide an adjustable bias on T9 for zero setting the DAC output. The 1mA FSD meter connected to the output is used to indicate converter operation.

Let us now turn to the complementary aspect of conversion — **analog-to-**

Tektronix...



Type 601:
5" Storage Display Unit

- Bistable storage
- 1-V Full-scale deflection factor for vertical and horizontal amplifiers
- Remote programming of display functions
- All solid-state

This unit permits stored displays of combined alphanumeric and graphic information from digital computers and other data transmission systems. The Tektronix-developed bistable Storage CRT used in the Type 601 eliminates costly memory devices for refreshing the information display. Built-in vertical and horizontal amplifiers permit versus T plots up to 100kHz for remote storage.

Type 611:
11" Storage Display Unit

- Flicker-free displays
- High resolution alphanumeric graphic display capabilities
- Write-through ability
- Remote programming of display functions

The Type 611 Storage Display Unit permits stored displays of combined alphanumeric and graphic information from digital computers and other data transmission systems. The 11 inch Tektronix-developed bistable storage CRT used in the 611 eliminates costly memory devices for refreshing the information display, provides high information density without flicker and with excellent resolution. A write-through feature provides ability to visually position the writing beam to any point on the CRT display area without disturbing previously stored information. Vertical format display area provides the same aspect ratio as a typewritten page.

Type 602:
5" Display Unit

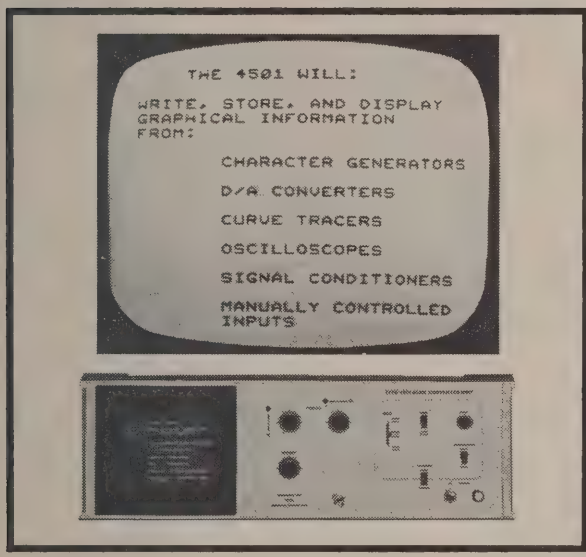
- 1-MHz X and Y bandwidth
- 100mV/cm X and Y deflection factors
- X-Y phase difference within 1° to 1-MHz
- Uniformly small spot size
- DC-coupled Z axis
- All solid-state

A compact, solid-state unit with excellent resolution providing accurate displays of information from X, Y and Z signal inputs. Application areas are: phase shifts and frequency ratios using Lissajous figures, graphic and alphanumeric displays from computers, high-resolution raster displays with intensity modulation and Y-T plots of amplitude versus time displays.

Five comprehensive units ready to engineer into your read-out system

S3076-L

can solve your display and storage problems!



Type 4501: Scan Converter Unit

- Bright hi-contrast display
- Composite displays
- Size limited only by monitor selection

The Type 4501 Scan Converter Unit converts information written on its 5-inch storage tube to composite video or modulated RF for convenient viewing on large-screen television monitors or receivers. Provision is made to select an output video signal that conforms with either 525-line, 60-field or 625-line, 50-field television standards. The display size is limited only by the choice of TV monitor or receiver. The bright displays achieved are ideal for individual or group viewing under high-ambient light conditions. A Tektronix 5-inch rectangular bistable storage tube is utilized as the memory device and provides resolution equivalent to 100 by 125 stored line pairs.

Type T4002: Graphic Computer Terminal

The Tektronix Type T4002 Graphic Computer Terminal is a completely self-contained, desk-top, information display system designed to facilitate rapid and efficient exchange of information between man and computer. Complete communication interaction is achieved through a solid-state, data-entry key-board and visual displays of high-resolution alphanumerics and graphics. A Tektronix developed, direct-view storage CRT is used as the display medium. High-density alphanumerics and complex graphics are presented without flicker or drift. Resolution achieved is equivalent to 400 x 287 line pairs.

Improved utilisation of the Terminal is provided by the addition of the optional 4901 Interactive Graphic Unit, which generates a bright, no parallax, crosshair cursor. The cursor is easily and accurately positioned with desk-top Joystick, and data points and instructions may be entered through the T4002 keyboard.

CONTACT YOUR TEKTRONIX FIELD ENGINEER FOR FURTHER INFORMATION AND ADVICE ON YOUR SPECIFIC NEEDS.

Distributed by Tektronix Australia Pty. Limited
N.S.W.: 80 Waterloo Road, Nth. Ryde 2113. Tel. 88 7066
VIC.: 25-27 Alma Road, St. Kilda 3182. Tel. 94 0229
S.A.: 128 Gilles St., Adelaide 5000. Tel. 23 2811

TEKTRONIX

S2076-R

digital conversion or "ADC." As one might expect, this aspect finds application wherever information in analog form must be fed to, or used to control, equipment designed to handle digital information.

A very practical application of ADC is the digital voltmeter or "DVM," in which the analog input is simply the voltage to be measured while the digital output is the number indicated by the readout display system.

Many other digital instruments are also applications of ADC. Examples are digital thermometers, pressure gauges, flowmeters, resistance bridges and Q meters.

There are many different ways of performing ADC—many more, in fact, than we can hope to discuss here. However, in general, all ADC's depend for their operation on an analog logic element usually called a **comparator.**

The function of an analog comparator is to deliver a digital output signal whose value depends upon the relative values of two analog input signals. This is illustrated in figure 5, which shows the logic symbol of a comparator together with the expressions which define its operation.

As may be seen, the digital output C is true (=logical 1) if the positive input A is larger than the negative input B. Conversely C is false (=logical 0) if B is larger than A. The comparator thus performs a comparison between A and B and gives the answer in binary form.

Figure 6 shows the circuit of the simple analog comparator used in the author's digital demonstrator unit. A brief look at this circuit may help in understanding how comparators operate.

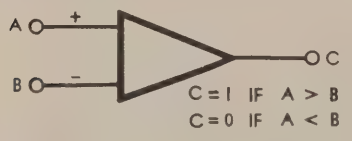
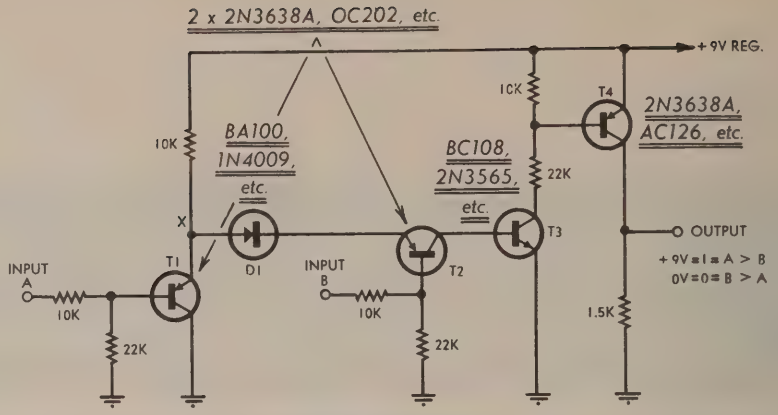
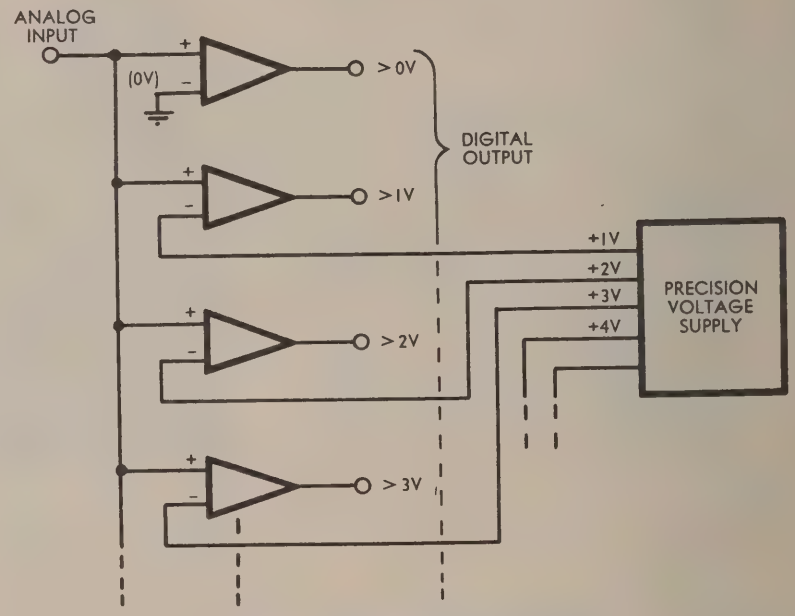


Fig. 5 THE ANALOG COMPARATOR



ANALOG COMPARATOR

Fig. 6



SIMULTANEOUS A-D CONVERTER

Fig. 7

In this circuit, T1 and T2 are input emitter followers, again of the "inverted" type to provide low offset. They share a common emitter resistor, T2 being coupled to the resistor via a diode D1, which serves to provide it with a small initial hold-off bias. The collector of T2 is coupled to the base of T3, which in turn couples to output transistor T4 to give a high voltage amplification.

Operation of the circuit is as follows: With both inputs at zero, T1 is saturated so that point X is only a few hundred millivolts positive with respect to ground. The turn-on voltage of D1 is such that the emitter of T2 is at near-zero potential under these conditions; thus T2 is cut off, despite the fact that its base and collector are both at ground. And with T2 non-conducting, T3 and T4 are similarly cut off and the output is also at ground (= logical 0).

The output will remain at logical 0 if a positive signal is applied to input B, because T2 will already be cut off. However, if B remains at zero and input is applied to A, transistor T1 will come out of saturation and the voltage at point X will rise. As soon as it rises sufficiently to overcome the turn-on voltage of D1, this voltage will be applied to T2.

With the B input at zero, T2 will saturate immediately this occurs, and will provide input current to T3. Hence T3 and T4 will be driven rapidly into saturation and the output will rise to +9V (logical 1). The circuit thus acts as a high-gain DC amplifier if we consider input A alone.

However, as soon as positive voltage is applied to input B, T2 is provided with hold-off bias which tends to counter the forward bias applied to its emitter via D1. A signal applied to B can thus be used to balance the effect of a signal at A and restore the output voltage to zero (= logical 0).

In order that the circuit may have minimum offset, diode D1 is selected so that with inputs A and B both at zero its turn-on voltage is just high enough to prevent T2 conducting due to the saturation voltage of T1. Thus T2 will tend to conduct immediately input A rises, and input B will be able to counter A on a 1:1 basis.

Under these conditions the circuit has very low offset and a very high gain, giving a small "aperture." The latter term describes the value of (A—B) or (B—A) required to change the output from logical 1 to 0 or vice-versa.

The diagram of figure 7 shows one way in which analog-to-digital conversion may be performed using analog comparators. The system shown is usually called "simultaneous ADC," as it provides a continuous and almost instantaneous digital output.

The **simultaneous ADC** consists of a series of analog comparators whose "+" or "A" inputs are connected in common to the analog input, and whose "—" or "B" inputs connect to a series of graduated precision reference voltages. The output of each comparator thus indicates whether or not the analog input is greater or less than the reference voltage concerned, and the comparator outputs as a whole give a digital representation of the magnitude of the analog input.

The output is in the form of what might be described as "cumulative decimal" notation, but this can be translated fairly easily into standard decimal, pure binary or BCD code as desired. The signal-handling capacity and resolution of the simultaneous ADC is more or less directly proportional to the number of comparators employed.

While the simultaneous ADC is extremely fast in operation, it tends to be rather costly as a result of the number of comparators and precision voltages required. For this reason it is used only in applications requiring maximum pos-

sible conversion speed with cost of little concern.

Naturally enough, in most applications, cost and its reduction are of vital concern. It is therefore not surprising that designers have found many alternative ADC methods to obviate the need for multiple comparators and reference voltages. Some of the more common of these methods will now be briefly discussed.

Figure 8 shows the logic diagram of the most economical ADC method, which is usually called "counter conversion." It may be noted that the ADC includes a DAC as one of its components, a feature possessed in common by most single-comparator ADCs.

The **counter-type ADC** consists of a counting register fed via an AND gate with pulses from a "clock" pulse generator. The second input of the AND gate connects to the output of an analog comparator, so that the comparator effectively controls the flow of pulses into the register.

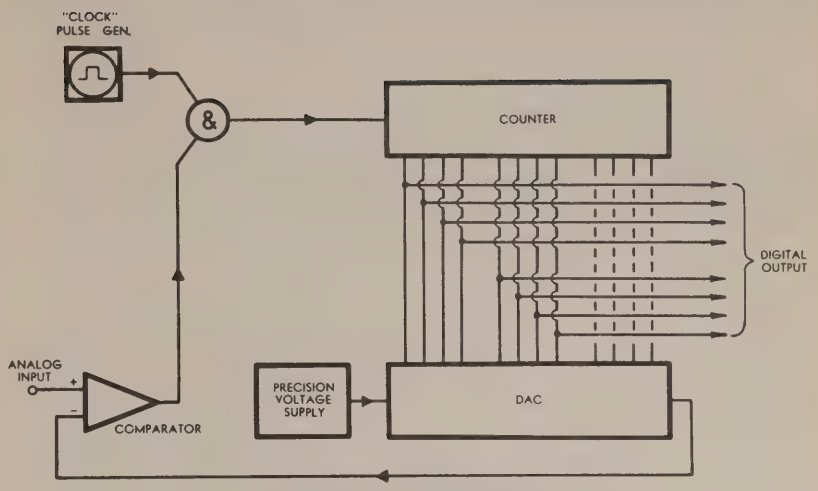
The analog input signal is applied to the + input of the comparator, and thus tends to open the AND gate and allow pulses to flow into the counter. When this occurs, the digital registration of the counter will rise and the analog DAC output will rise also.

As the DAC output is connected to the — input of the comparator, the latter will be forced to close the AND gate as soon as the DAC output voltage equals and exceeds the input voltage. Counting thus stops when the digital registration of the counter is equivalent to the analog input voltage, giving the required analog-to-digital conversion.

It may be seen that the counter-type ADC uses a counter and DAC to generate a rising analog "staircase" waveform which forms the reference voltage for the single comparator. In effect, it "tries" a series of increasing reference voltages at the negative comparator input, and stops when the reference balances the analog input.

The counter-type ADC tends to be rather slow in operation as it must build up a balancing reference voltage by cumulative counting. It has the additional disadvantage that it can only operate continuously for increasing input signals. To cope with decreasing signals at all it must be arranged to "sample" at periodic intervals by resetting the counter and forcing it to reach a new balance.

As a result of its shortcomings, the counter-type ADC is restricted mainly to applications where slow-speed sampling-type conversion is required. Probably the most common application is in low-cost digital voltmeters.



COUNTER-TYPE A-D CONVERTER

Fig. 8

The **"continuous" ADC** is a development from the counter type. In place of the simple counting register the continuous converter employs a bi-directional register, and uses two analog comparators to control both pulse input to the counter and counting direction. It is thus able to give a continuous conversion, as it can follow both upward and downward variations in the analog input.

Although the continuous ADC lacks one of the shortcomings of the counter type, it retains the disadvantage of slow following speed with respect to large changes in input. Thus it, too, tends to be restricted to slow speed applications although less so than in the case of the counter type.

In general, ADCs designed for higher operating speeds adopt "short-cut" means to generate the comparison voltage, rather than generate the voltage using simple or bidirectional counting. For example, there is the "sequential approximation" ADC, which achieves a rapid balance by commencing with a large reference voltage step and then applies smaller and smaller corrections to this depending upon the state of the comparator. Very high-speed ADC's use even more complex balancing techniques and sometimes use a combination of a number of techniques.

As mentioned earlier in this chapter the space is not available here for a complete description of all methods used for performing analog-to-digital conversion. It is hoped that the foregoing will give some insight into the principles involved; readers interested in pursuing the topic further will find additional information in the references listed below.

SELECTED BIBLIOGRAPHY

MILLMAN, J., and TAUB, H., **Pulse, Digital and Switching Waveforms**, 1965. McGraw-Hill Book Co., New York. Ch's 7, 8, 9, 18.

PEARMAN, C. R. and POPODI, A. E. "How To Design High-Speed D-A Converters," in **Electronics**, V. 37, No. 8 (Feb. 21, 1964).

TATRO, R. D., **Digital to Analog Conversion**, Sprague Technical Paper No. TP-66-5. (Available by request on company letterhead to D. R. Johnston and Co. Pty. Ltd., Suite 14, Stanhill 33, Queens' Road, Melbourne.)

THE DIGITAL LOGIC HANDBOOK, 1966-67 Edition. Digital Equipment Corp., Maynard, Massachusetts. (Available by request on company letterhead from Digital Equipment Australia Pty. Ltd., 89 Berry Street, North Sydney, N.S.W.)

Chapter 9

TIMING and CONTROL

Signal shaping—the Schmitt trigger—input circuits for mechanical contacts—"clock" pulse generators—the astable multivibrator, unijunction oscillator, quartz oscillator and R-C oscillator—enabling and disabling—pulse delay circuits—the monostable multivibrator or "one-shot"—beginning timing or pulse "stretching"—end timing—synchronisers—level, start-stop and pulse synchronism.

Digital circuitry deals with electrical signals representing numerical information, as we have seen in preceding chapters. In general it is true that the electrical signals concerned can occupy only two possible voltage or current levels, and they can represent numerical values either by static voltage or current levels or by dynamic level transitions. The latter are conveniently subdivided into single transitions of "steps" and reflexive transitions of "pulses."

Thus in terms of static levels, a binary or logical "1" may be represented by a DC voltage of a specified value applied between a wire and earth, with "0" represented by a second voltage value. The actual values used are assigned arbitrarily; thus "1" may be represented by a convenient positive or negative voltage, or by zero volts, while "0" may be represented by any other convenient value. The terms "positive logic" and "negative logic" are used to signify whether "1" is either more or less positive than "0," respectively.

In terms of transitions, "1" may be represented by either a positive- or negative-going step, or by either the leading or trailing transition ("edge") of a pulse which may itself be either positive- or negative-going. In both step and pulse cases "0" may be represented by either a step or pulse of opposite polarity, or by simply the absence of the "1" transition.

In order that the various digital circuit elements can operate effectively from and upon signal transitions, the latter must usually be arranged to take place at a standard rate determined by the characteristics of the circuit elements. Transitions which are faster or slower than the standard rate will usually cause faulty circuit operation.

Since the contemporary design emphasis is on increasingly faster equipment operation, this usually means that transitions are kept as fast as the circuit elements will permit—and the elements themselves are continually improved to permit faster and faster operation.

Thus it may be seen that, for effective operation of digital circuitry, the signals with which it deals must satisfy two criteria: They must have only two values, and they must make any significant transitions between these values at a suitably rapid rate. In order to meet these criteria, signals to be processed by digital circuitry must usually be "shaped" or "squared up."

Usually the same operation must also be performed at various stages throughout the digital processing in order to maintain the signal waveforms, despite degradation introduced by the inevitable stray capacitance and inductance of the circuitry.

Almost invariably the squaring-up operation is performed by bistable trigger circuitry based on a circuit developed in 1938 by O. H. Schmitt. Figure 1 shows simple transistor circuits for such **Schmitt trigger** waveform shapers,

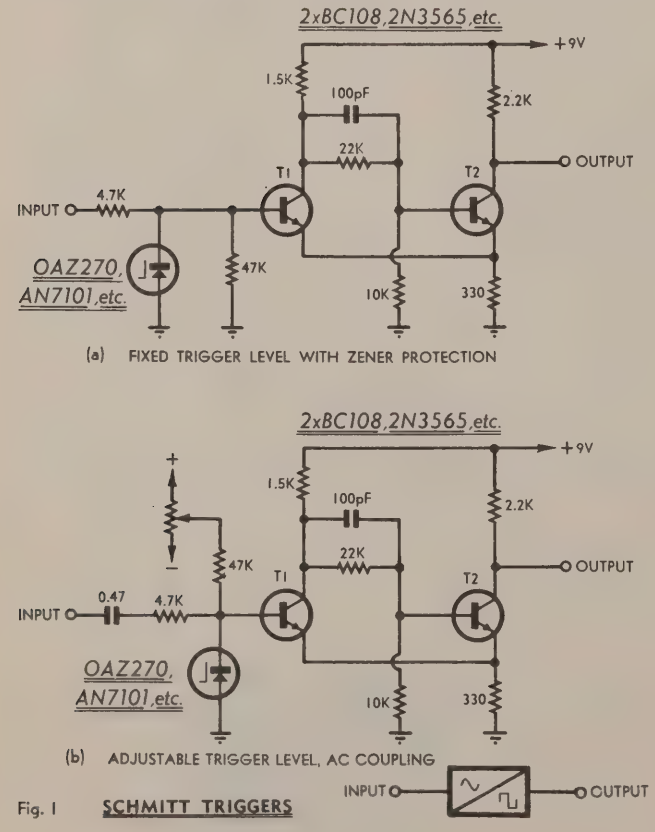


Fig. 1 SCHMITT TRIGGERS

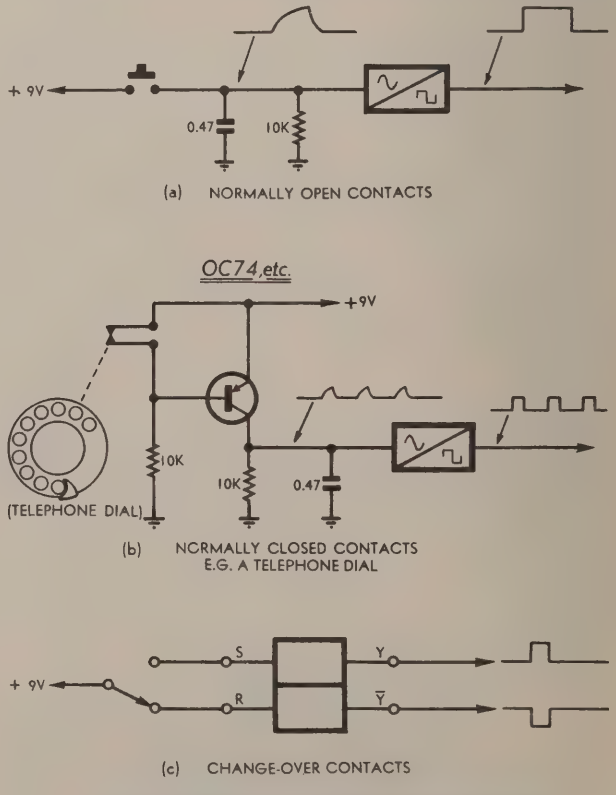


Fig. 2 INPUT CIRCUITS FOR MECHANICAL CONTACTS

together with the usual logical symbol for these elements.

Figure 1 (a) shows the circuit of a fixed level trigger. As may be seen, it consists of two transistors sharing a common emitter resistor, with coupling whereby the base of T2 is supplied with approximately 1/3 of the collector voltage of T1.

If the input to the circuit is at zero voltage with respect to earth, T1 has no forward bias and is cut off. Its collector thus rises to almost +9V, carrying the base of T2 to approximately +3V. Transistor T2 therefore conducts heavily, and the output terminal is clamped at a low voltage level.

The circuit remains in this stable state for all values of input voltage which leave the base of T1 at a voltage lower than that across the 330ohm emitter resistor. However, whenever the input voltage raises the base of T1 significantly above this critical level, the circuit switches over extremely rapidly to the reverse state with T1 conducting and T2 cut off. The rapid switchover is produced by the large degree of feedback present between the two transistors, assisted by the 100pF "speedup" capacitor between T1 collector and T2 base.

If the input voltage maintains the base of T1 above the triggering level the circuit remains stably in the reverse state, with the collector of T2 at almost +9V in contrast with its former near-zero value. The circuit does not return to its former state until the base of T1 returns to value **slightly below** the triggering-on value.

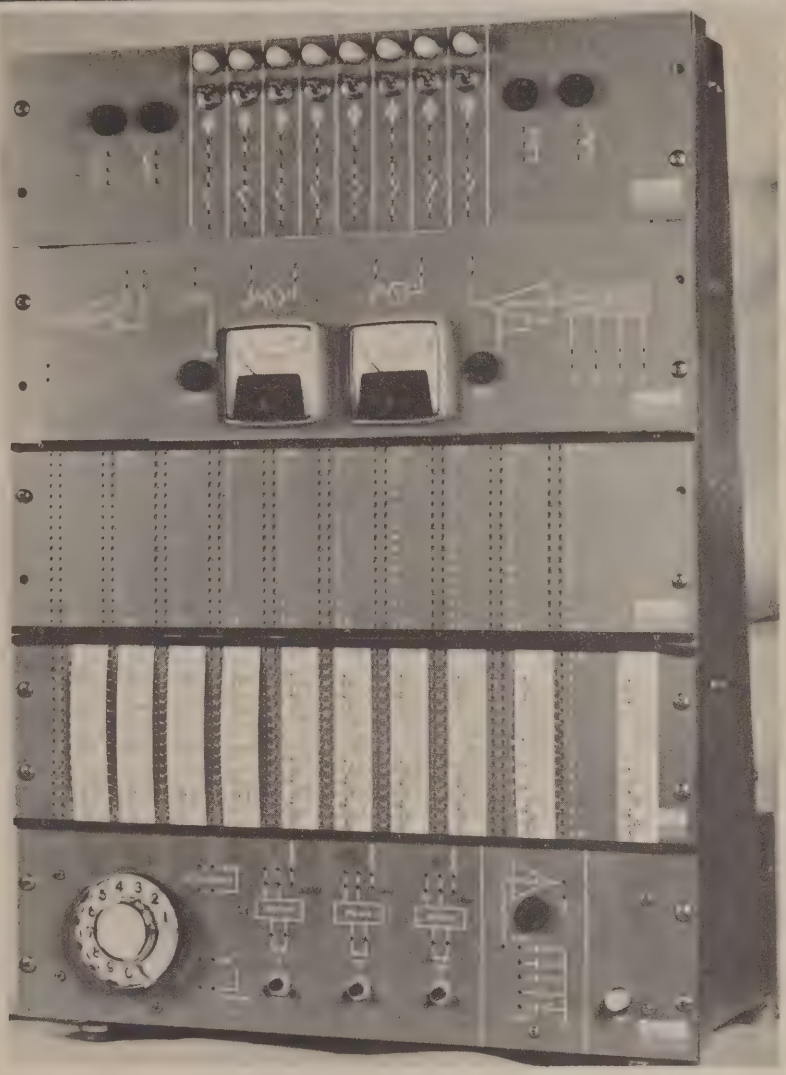
The slight difference between the triggering-on and triggering-off levels is termed the **hysteresis** of the trigger. It may be reduced to zero or even made negative by suitable modifications to the basic circuit; when made negative the circuit becomes astable and oscillates continuously. However, for many applications a small hysteresis is not unduly embarrassing, and the basic circuit shown is thus found quite satisfactory.

It should be fairly evident that a signal applied to the trigger input will produce an output having only two possible levels (here about +3V and +9V respectively), and with the transitions between these levels occurring very rapidly (rise time is about 400 nanoseconds, fall time about 100 nanoseconds). The trigger is thus very suitable for producing a squared-up version of almost any sort of input signal, whether the latter be a sinewave, sawtooth, triangular wave or a degraded ("rounded") square wave.

To protect the circuit from damage due to accidental application of high or negative voltages to the input terminal, a 6V zener diode is shunted from the base of T1 to earth. The forward and zener characteristics of the diode ensure that the transistor base cannot be taken lower than about —0.5V nor higher than +6V.

The circuit of figure 1(b) differs from that in (a) in that the input is coupled to the base of T1 via a DC blocking capacitor; also in that a variable DC bias may be applied to the base to permit adjustment to the level of the input signal waveform at which the circuit triggers. These modifications allow the trigger both to ignore any DC level possessed by the input signal and to be adjusted to trigger from a specified portion of the signal, should these functions be required.

DIGITAL DEMONSTRATOR, DESIGN TOOL



The Digital Logic Laboratory, a commercial demonstrator and design breadboarding unit of similar type to the unit being described in these articles; although it provides considerably larger numbers of higher-performance logic elements. It is made by the Digital Equipment Corporation of Maynard, Massachusetts, manufacturers of a range of small- to medium-sized high-speed digital computers. The Laboratory employs the same high-performance "Flip-Chip" (T.M.) modules used in the firm's computers and is thus eminently suitable for breadboarding of sophisticated high-speed digital designs.

The laboratory is available in three versions offering various degrees of functional flexibility. Pictured is the most elaborate version, which features nine multiple-gated flip-flops, four quadruple NOR gates, a quadruple NAND gate, two dual delay one-shots, a dual gated-pulse amplifier, eight indicating lamps and drivers, a D-A converter panel, a power supply and signal generator panel and some 16 boxes of various length patching cords. All three versions are accompanied by a workbook which gives a graded series of tuitional experiments and demonstrations.

Enquiries regarding the laboratory may be directed to Digital Equipment Australia Pty. Ltd., at 89 Berry Street, North Sydney, N.S.W.

It is necessary to make special arrangements where digital input signals must be derived from mechanical contacts and switches, even though at first sight the signals produced by such contacts appear to be in ideal "on-off" digital form. The reason for this is that almost all mechanical contact systems suffer to some degree from "bounce" —a defect wherein the contacts do not make or break cleanly and positively.

If it were not rendered ineffective, bounce would cause defective circuit operation because it adds spurious transitions to the original information. Thus if a set of contacts bounce twice in closing, they effectively perform three successive open-closed transitions instead of the intended single transition.

Figure 2 (a) shows the simplest method of suppressing bounce for contacts that are normally open, such as those in a push-button. Here a capacitor and resistor are used to integrate the voltage from the contacts, "lumping" any rapid bounce pulses in with the main transitions. A Schmitt trigger is used to square up the integrated pulses and produce the final useful signal.

Where normally closed contacts are

used, it may prove convenient to incorporate an inverting transistor as shown in figure 2 (b). Here the contacts used for illustration are those provided on a telephone dial mechanism for pulsing. As before, the pulses are first integrated and then squared-up using a Schmitt trigger.

Where change-over contacts are used for digital input the arrangement of figure 2 (c) is often used. Here a simple S-R flip-flop is used to act as a "buffer." As may be recalled, such a flip-flop becomes insensitive to succeeding signals applied to an input once it has switched to the state dictated by the first signal to arrive at that input. Thus multiple pulses produced by contact bounce are ignored, although the flip-flop outputs switch rapidly to produce an output having "fast" transitions.

Operations performed by digital circuitry must often be either synchronised with one another or performed in a specific time sequence. Specialised circuit elements and configurations are used to ensure that the required time relationships are achieved; these circuit elements and configurations will now be examined in turn. They comprise "clock" pulse generators, delay and timing elements and synchronisers.

Clock generators are circuit elements which generate pulses suitable for use throughout a digital circuit as time reference markers. Almost any circuit which generates fast rectangular pulses of reasonably stable width and repetition rate is suitable for use as a clock in many digital circuits, although in some applications it is necessary to use an ultra-stable clock generator of complex design.

Perhaps the simplest clock is provided by the astable multivibrator, as illustrated by the simple circuit of figure 3 (a). Here two transistors are interconnected so that they alternately and complementarily saturate and cut off each other, with a repetition rate determined by the coupling capacitors C and the circuit resistances.

When operating a circuit of this type produces near-rectangular pulses of easily controlled repetition rate; although the stability is not particularly good. The diodes D1 and D2 proect the transistor base-emitter junctions from reverse voltage breakdown during the peaks of the turn-off pulses applied by the opposite transistor.

Of similar performance to the astable multivibrator is the unijunction relaxation oscillator, of which a simple example is shown in figure 3 (b). Here capacitor C charges repeatedly to the conduction potential of the unijunction, which produces small pulses across its base-one resistor when it conducts.

The repetition rate is determined by the charging resistance R, and is typically adjustable over a range from less than 1Hz to more than 100KHz. The unijunction output is squared-up and amplified to provide a full amplitude signal by a Schmitt trigger.

Figure 3 (c) shows a simple circuit for a quartz crystal clock, used where the clock pulses are required to be of a stable repetition rate. In some applications the crystal would be operated in a constant-temperature oven to give even greater stability.

Crystal clocks are usually operated between 100KHz and 5MHz. In general, a Schmitt trigger is used to ensure a rectangular output waveform, as shown.

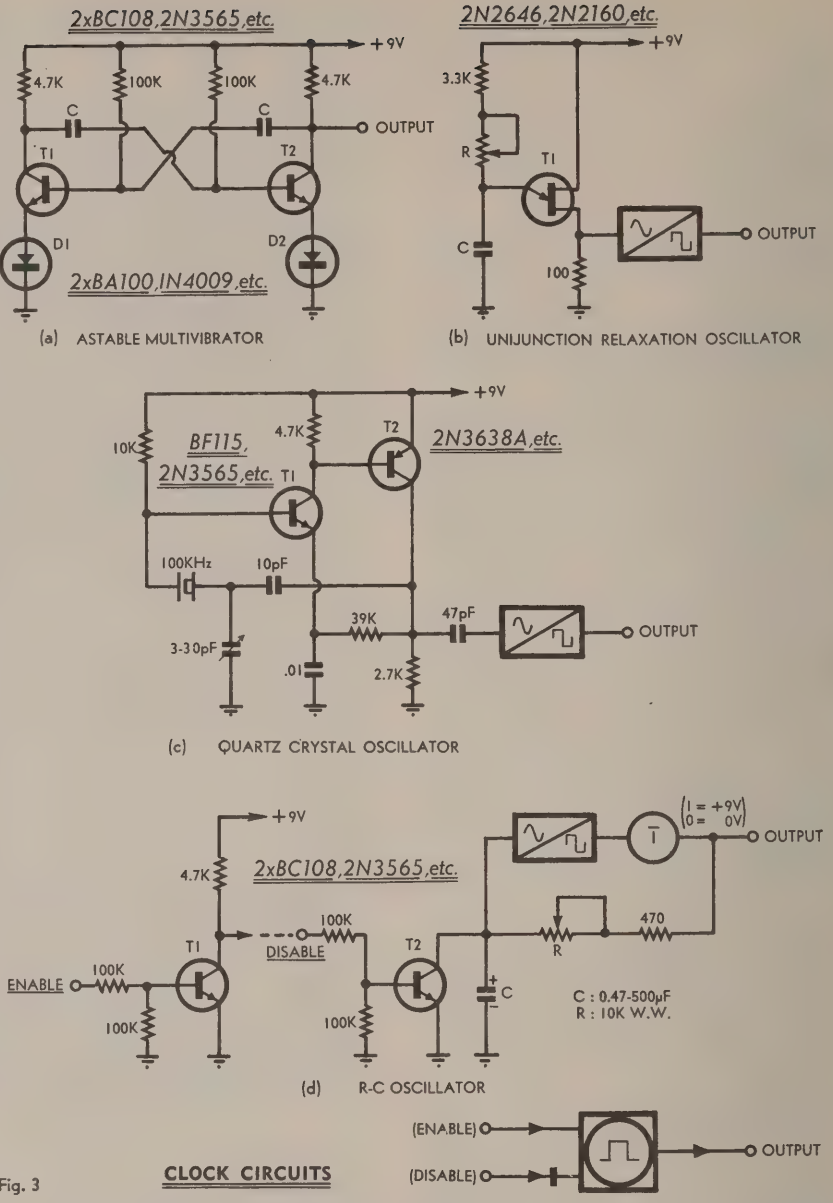


Fig. 3

CLOCK CIRCUITS

A Schmitt trigger and an inverter may be used to form a simple R-C clock oscillator which is often used, both because of its clean output waveform and because it may be easily controlled by an external level signal. The basic arrangement is shown in figure 3 (d).

Neglecting transistors T1 and T2 for the moment, it may be seen that resistor R and capacitor C form a delaying feedback circuit around the trigger and inverter. Thus with the trigger output at "0" the inverter output is at "1," and C charges through R.

As soon as the voltage across C reaches trigger-on level, the trigger and inverter change state and the output switches to "0." Then C starts to discharge through R, and does so until its voltage falls below the trigger-off level. When this occurs the trigger and inverter change state again, and the cycle repeats itself. The output signal produced is a clean rectangular wave whose repetition rate is easily adjusted by varying R and C.

The operation of the circuit can be stopped by applying a +9V level signal to the series base resistor of transistor T2, as this causes T2 to conduct to saturation and prevent C from charging. Thus a level signal may be used to **disable** the clock when it is not required.

By employing an additional transistor T1 as an inverter the control level signal may be used to **enable** the clock rather than disable it — i.e., the clock will be allowed to operate only in the presence of the control signal rather than only in its absence.

R-C clocks of the general type shown in figure 3(d) are widely used in digital circuitry to supply pulses from less than 100Hz to over 10MHz.

Delay elements are used in digital circuits to introduce signal time delays where these are essential for ensuring the correct sequence of logical operations. The delays which may be required for this vary over a very large range; in high-speed logic they may be as small as a few nanoseconds, whereas in special industrial applications they may be as large as 30 minutes.

Very short time delays can often be provided by standard logic elements, as transistors and diodes themselves introduce delays due to P-N junction storage and transition time. In high-speed logic circuitry such delays may provide all that is required.

Where controlled delays ranging from a few nanoseconds to a few microseconds are required, the most com-

monly used element is the electrical **delay line.** This consists of either a length of transmission line — usually made from a special type of coaxial cable — or an electrical equivalent of such a line constructed from inductors and capacitors.

The former type is termed a "distributed constant" element, while the latter is termed a "lumped constant" element. Both types produce a time delay for pulses and steps by presenting them with a combination of series inductance and shunt capacitance, with the values of these constants determining the delay time.

The diagram of figure 4(a) shows the basic circuit of a fairly common arrangement using a distributed-constant line with a terminating and inverting transistor. The overall delay of the element consists of that of the line itself together with the delay of the transistor.

Delays longer than those conveniently obtainable with delay lines are often obtained using an R-C integrator and Schmitt trigger, as illustrated in figure 4(b). Here the R-C time constant effectively delays the rise and fall of the signal transmitted to point "A," so that the trigger switches on after a time t and off after a similar delay.

Where long delays or very precise delays of moderate length are required, the delay element most commonly used is the **monostable multivibrator,** also called the "one shot" or the "unistable." This is illustrated by the simple circuit of figure 4(c).

Here transistors T1 and T2 form the monostable proper; T3 is a differentiating inverter whose purpose will be explained in a moment.

As may be seen, the collector of T1 is directly coupled to the base of T2 as in a normal flip-flop. However, in place of the second direct coupling from T2 collector to T1 base, there is simply a coupling capacitor C giving AC coupling only.

With the input to the circuit at 0V, T1 is cut off and its collector is at approximately +9V. Hence T2 is provided with heavy forward bias, and accordingly saturates with its collector at near-zero potential. Under these conditions capacitor C is essentially uncharged.

As soon as the input terminal is taken to logical "1" (here +9V) transistor T1 is driven into saturation. This removes the bias from T2, and accordingly the latter switches off. Its collector voltage thus tends to rise sharply, whereupon capacitor C is forced to commence to charge to +9V.

Because the base-emitter junction of T1 and the series emitter diode D1 represent a low resistance path, the bulk of the capacitor charging current flows through the transistor as forward bias. Thus T1 is held in the saturated mode until the charging current through C falls to a value below the minimum saturation bias level.

If the input signal has fallen to zero by the time the charging current falls to this minimum figure, T1 rises out of saturation and the cascaded gains of T1 and T2 switch the circuit rapidly back to its original state. Capacitor C is discharged through T2 and R1, driving T1 rapidly to cutoff; diode D1 protects the latter from base-emitter breakdown.

As may be seen, for input signals which end before C has charged the circuit switch-off time t is effectively

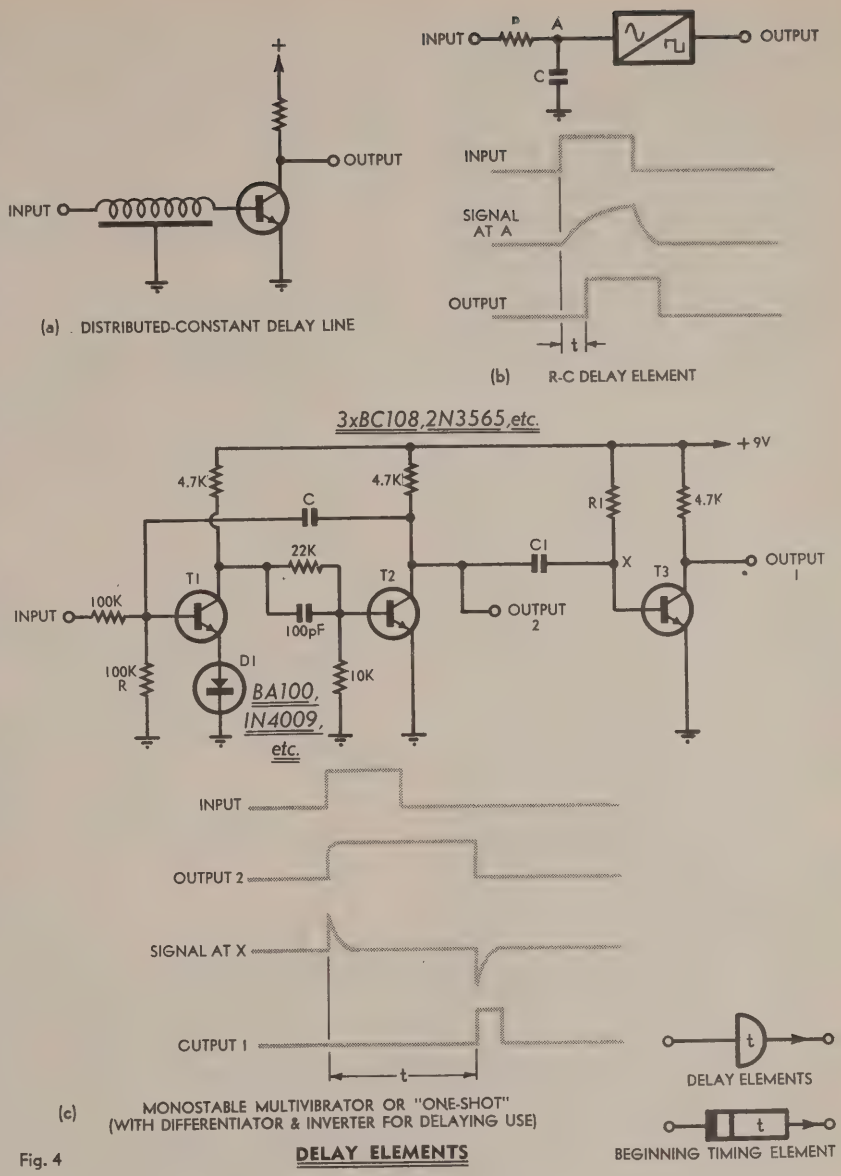


Fig. 4 **DELAY ELEMENTS**

independent of the switch-off time of the input signal; it is determined by the value of C and its associated charging resistance. The monostable is thus basically a "pulse stretcher."

Note that for input signals which do **not** end before the monostable would normally switch off, the circuit will act roughly like a trigger. Its output ("output 2") will simply follow the input for both switch-on and switch-off, contributing only power amplification and a certain amount of squaring-up.

To use the monostable as a pulse delay element it is usually combined with a differentiator and inverter, as shown. Capacitor C1 produces at point "X" a differentiated version of the monostable output; while transistor T3 inverts, amplifies and squares-up the second differentiated pulse, that corresponding to the monostable switch-off transition. Hence the signal at the "output 1" terminal is a pulse, delayed by time t with respect to the start of the input pulse.

The direct or "stretched" output of the monostable may be used for **timing** applications, as distinct from those requiring a delayed pulse output. Used in this way, the monostable is regarded as a "beginning timing" element rather than a delay element, its function then being interpreted as producing a 1—0 transition at time t after the beginning of the input pulse.

Note from figure 4 that the logic symbol for a delay element is different from that for a beginning timing element, so that the monostable would be represented by either one symbol or the other depending upon the interpretation placed upon its operation in the application concerned.

Besides the beginning timing element there are a number of other elements used for logical timing as distinct from pulse delay; however, space limitations prevent us from considering more than an example of one other type. This is the **end timing element,** whose function is to generate a pulse of desired length which commences at the **end** of an input pulse.

Figure 5 shows the circuit for a simple end timing element, together with its principal operating waveforms. As may be seen, it is nothing more than the differentiating inverter stage used to convert the monostable into a pulse delay element. When used alone it simply produces an output pulse of width t determined by the values of C and R, the pulse commencing synchronously with the end of the input pulse.

Arrangement of logical elements to form **synchronisers** is required whenever a randomly occurring signal is required

to produce a synchronous effect—i.e., an effect which is properly timed with respect to the clock pulses used by the equipment.

Consider the case where a push-button is to be used to operate a gate to let clock pulses through to a counter. The level signal provided by such a push-button is a "random" one, because the operator can push and release the button at any time. Thus there is a definite chance that the gate may be opened and closed during clock pulses, which will result in fractional pulses being passed to the counter. And fractional pulses can often cause either circuit malfunction or timing errors, so that this arrangement as it stands is not satisfactory.

Now consider in contrast the logic diagram shown in figure 6(a). Here the gate is not controlled directly by the level signal derived from the random input, but by the Y output of a J-K flip-flop. The level signal is merely used to control the levels at the J and K inputs of the flip-flop, while the trigger input is connected to the same source of clock pulses connected to the gate.

This arrangement operates in the following way: With the level signal at logical "0," the flip-flop remains reset, with its Y output also at logical "0." The clock pulses cannot switch the flip-flop over into the set state because the J input remains at logical "0," and thus the AND gate remains closed.

If the level signal is taken to logical "1," for example by pressing a push-button, the flip-flop J input will thus go to "1," while the K input will go to "0." The flip-flop will thus change its state—not immediately, but on the arrival of the **first clock pulse** following the change of levels at the J and K inputs.

Although the flip-flop will change state on the arrival of this first clock pulse, it will take a certain time to switch over. This means that the AND gate will not actually open until after the first clock pulse has ended. The first clock pulse to pass through the open gate thus will be the **second** occurring after the random level signal transition, and this pulse will be passed completely.

Closure of the gate is performed in a similar fashion. Return of the level signal to "0" simply returns the inputs J and K of the flip-flop to their original values ("0" and "1" respectively). The next clock pulse following this random event thus is able to switch the flip-flop back to the reset (Y=0) state, which closes the gate—after the clock pulse has already passed through.

It may be seen that the arrangement of figure 6(a) permits a random level control signal to operate a pulse gate synchronously—i.e., so that turn-off and turn-on are at a specific time in relation to the clock pulses. This logical configuration is accordingly called a **"level synchroniser."**

A second type of synchroniser is the **"start-stop"** synchroniser, which is illustrated in figure 6(b). Here the gate is opened synchronously under the command of randomly operated "start" and "stop" push-buttons—or, alternatively, by random pulses performing the same function.

The configuration is basically similar to that of (a) except that the J and K inputs of FF1 are controlled not by the control signals but by a second S-R flip-flop FF2. The second flip-flop acts as a storage element or "buffer," storing the random command signals from either the push-buttons or their equivalent pulses until the arrival of the next command signal. In the case of the push-buttons flip-flop FF2 also acts as a bounce suppressor of the type shown in figure 2(c).

As with the level synchroniser the start-stop synchroniser opens the gate just after the first clock pulse to occur after the "start" command, and closes it again just after the first clock pulse to arrive after the "stop" command. Fractional pulses are therefore avoided.

A third type of synchroniser configuration is shown in figure 6(c). This is the **"double-pulse"** synchroniser, which permits a single random pulse to open a gate and let through **two** successive clock pulses. As double pulses are used quite frequently in digital equipment, this configuration is quite an important one.

Operation is as follows: Initially, both FF1 and FF2 are reset (Y=0) and the gate is closed. When a random control pulse appears, it thus sets FF2 to apply a "1" and a "0" respectively to the J and K inputs of FF1. (It is assumed here that the flip-flops used are those described in the fourth article in this series, in which the J and K inputs use the negative logic convention while the outputs use the positive convention. Hence with the interconnections shown, "1" at the Y output of FF2 produces "0" at the K input of FF1, and so on.)

The first clock pulse to occur after this random switch-over of FF2 thus is able to set FF1 to Y=1; thus, as before, the gate opens just after this clock pulse, and the next clock pulse passes through the gate. However, when it does so, it passes not only to the subsequent circuitry but also to the pulse reset (PR) input of FF2. Accordingly FF2 is reset to Y=0.

As a result, the J and K inputs of FF1 are restored to their original values of "0" and "1," so that the next clock pulse after the one which first passes through the gate (i.e., the third pulse from the random control pulse) not

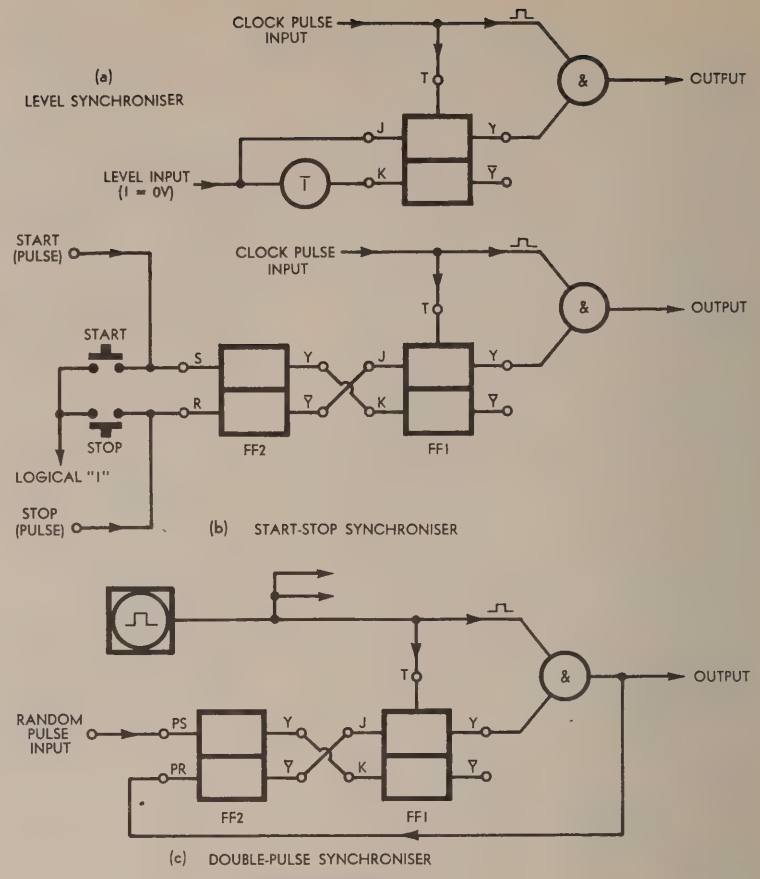
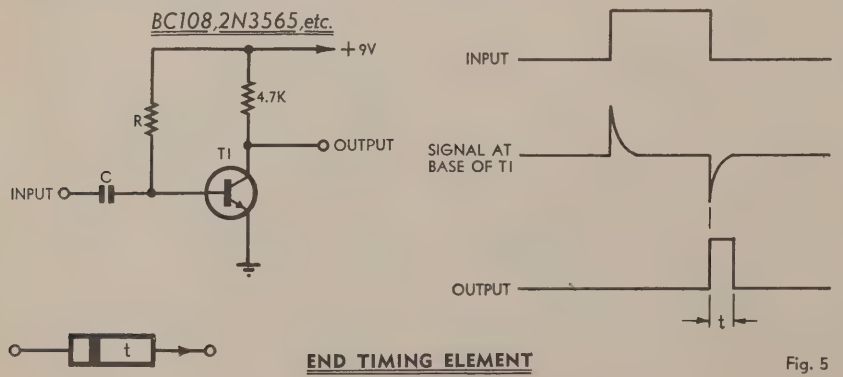


Fig. 6 **SYNCHRONISERS**



END TIMING ELEMENT Fig. 5

only passes through the gate but also resets FF1 to close the gate immediately after doing so.

In case the foregoing is found by the reader to be confusing, the following step-by-step analysis may make the sequence of operations clear:

1. Both FF1 and FF2 reset, gate closed.
2. Random command pulse sets FF2.
3. Next clock pulse to occur (CP1) sets FF1; gate thereby opened just after CP1 itself ends.
4. Succeeding clock pulse CP2 passes through gate, also resets FF2.
5. Third clock pulse CP3 passes through gate, also resets FF1 to close gate immediately afterward.

It may be seen that only two clock pulses pass through the gate (CP2 and CP3), and that as with the other synchroniser configurations these pulses are "whole." The double-pulse synchroniser thus permits a single randomly occurring pulse to initiate the propagation of two successive accurately timed clock pulses, which may in turn be employed for synchronous operations of gates, flip-flops, and so on.

An important application of the double-pulse synchroniser is in the gating and control circuitry of a digital timer/frequency meter, which is one of the digital instruments discussed in the next chapter.

SELECTED BIBLIOGRAPHY

DAKIN, C. J., and COOKE, C.E., **Circuits for Digital Equipment,** 1967. Iliffe Books Ltd., London.

DEAN, K. J., **An Introduction to Counting Techniques and Transistor Circuit Logic,** 1964. Chapman and Hall Ltd., London.

LEDLEY, R. S., **Digital Computer and Control Engineering,** 1960. McGraw-Hill Book Co., New York.

THE DIGITAL LOGIC HANDBOOK, 1967 Edition. (Available by request on company letterhead from Digital Equipment Australia Pty. Ltd., 89 Berry St, North Sydney, N.S.W. 2060.)

Chapter 10

DIGITAL INSTRUMENTS

The basic counter—over-ranging, overflow and display storage—start-stop and batch counters—time period counters—multiple period averaging—digital frequency meters—frequency ratio measurement—extending the frequency range—digital voltmeters—ramp, integrating and potentiometric types—digital phasemeters—digital thermometers.

Using the content of the preceding chapters as background we should now be in a position to look at the basic operation of the more common digital measuring instruments. The discussion will commence with the various types of counter, as the counter in one form or another is the basis of almost all digital instruments.

Figure 1(a) shows the general logic layout of the basic digital counter. It consists mainly of a cascaded series of decade counting circuits, together with the appropriate decoding and readout circuitry.

In most modern general-purpose counters the counting decades would consist of groups of four flip-flops or other bi-stable elements, as shown. Typically the flip-flops would be gated R-S or J-K types and would be arranged for ripple-carry counting in a suitable BCD code.

Alternatively, the decades may consist of gas-filled or vacuum (trochotron) decimal counting tubes, or perhaps of flip-flops or other bi-stable elements arranged in the normal or twisted ring counter configuration. As we have seen in earlier chapters the choice of counting device and configuration is influenced by a variety of factors, including the required counting speed, cost, decoding complexity, the readout system available, and so on.

Note that it is by no means necessary or even desirable to use the same devices or configuration in all the decades. In fact it is quite common to employ different devices and configurations, in order to take advantage of the important fact that each succeeding decade counts at one-tenth the speed of its predecessor.

Thus for the input decade of a very high-speed counter it may be necessary to use high-speed tunnel diodes in a twisted ring counter configuration, whereas for the second and later decades it may be sufficient to use a standard four flip-flop ripple-carry configuration. Some low-cost instruments even use mechanical pin-and-cog counters for the final decades, as these usually count at a very low rate.

A separate counting decade, decoder and readout system are required for each fully significant digit of the total count, as shown. The first and fastest decade handles the **least** significant digit (LSD), the second decade the **next** significant digit (NSD), and so on to the final and slowest decade handling the **most** significant digit (MSD).

As an addition to the basic array of counting decades many digital instruments provide a further incomplete decade or (more usually) a single flip-flop, to give a worthwhile extension of counting capacity at a relatively low additional cost. This is called **"over-ranging."**

Illustrated in figure 1 (a) is the usual over-range system using a single additional flip-flop (FF1). The flip-flop is a standard gated R-S or J-K type arranged for complementary-mode operation, and is fed with the output from the MSD decade counter. Driver amplifiers connected to the two output terminals are connected to two lamps illuminating "O" and "1" symbols respectively; no decoding is required.

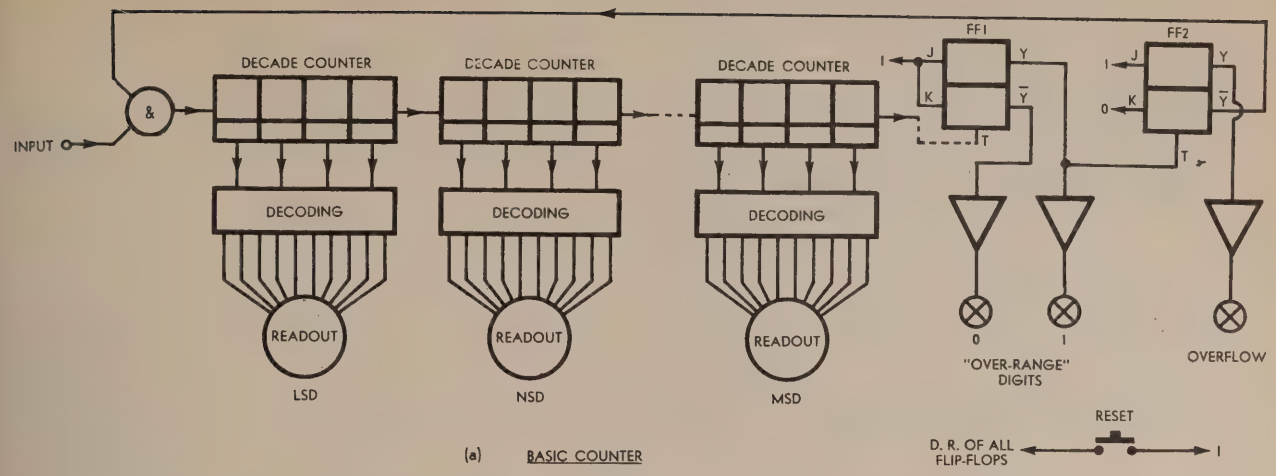
A little reflection should show that the addition of the single over-range flip-flop **doubles** the range of the counter. Thus if there are four full decades, which would normally be able to count from 0-9,999, the capacity with over-range will be 0-19,999.

Regardless of whether or not they provide over-ranging, most practical counters and digital instruments incorporate circuitry designed to stop the counting operation and give a suitable indication on the occurrence of a situation known as **"overflow."**

The overflow situation is simply that in which the full counting register has filled to capacity and returned to zero registration — in identical fashion to an auto odometer or mileage counter when it has gone "around the clock."

For example, an instrument with four decades and a single flip-flop over-range will fill completely on the arrival of 19,999 input pulses, and will over-

The Hewlett-Packard 3450A, a digital volt-ohm meter which attains a very high accuracy and resolution using a sophisticated dual-slope integration technique. (Courtesy Hewlett-Packard Australia Pty. Ltd.)



(a) BASIC COUNTER

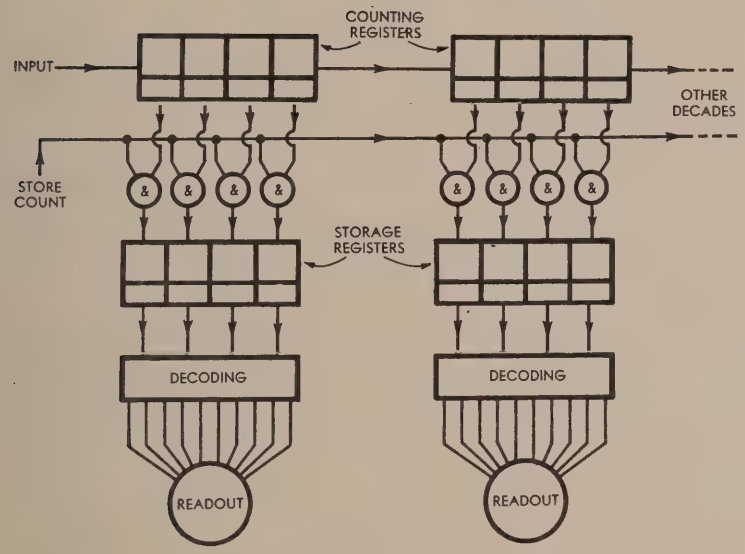


Fig. 1 (b) STORAGE-TYPE DISPLAY

Above and at left are diagrams used to explain the operation of the basic digital counter configuration.

flow on the 20,000th pulse — its registration will return to "00000."

In high-speed counting it is particularly essential that overflow be detected and the instrument be stopped when it occurs, because unless this is done the registration displayed when counting stops will be ambiguous. Thus an instrument of the capacity mentioned above and lacking overflow detection circuitry would display a registration of "16230" not only after the arrival of 16,230 pulses but also after the arrival of 36,230 pulses, 56,230 pulses, 76,230 pulses, and so on, depending entirely upon the number of times the register had overflowed.

To ensure that the registration displayed is completely unambiguous it is simply necessary to arrange that the input to the instrument is disconnected and locked out immediately the register overflows. For convenience it is usual to provide also for activation of an "overflow" indicator, in order that the operator be made aware of the situation and given the opportunity to change ranges, reduce the sampling time or make other arrangements to prevent the overflow from recurring.

Reference once more to figure 1(a) will show how overflow detection and lock-off are usually provided. A further flip-flop (FF2) is involved, with its T input connected to the output of either the MSD decade or the over-range flip-flop if the latter is used.

The Y output of the additional flip-flop connects via a driver amplifier to an indicator lamp marked "overflow," while the complementary output connects to an AND gate which is in series with the input to the register.

The overflow flip-flop is reset along with all the other elements of the counter — including the over-range flip-flop if this is used. Thus during normal counting the overflow lamp remains extinguished, while the AND gate at the input to the counter remains open to admit input pulses. However if the counting registers overflow, this will be indicated by a **1-to-0 transition** at the output of the MSD decade or over-range digit. Hence FF2 will set to $Y=1$, the overflow lamp will light and the input AND gate will close to lock out the input pulses.

It may be seen that ambiguous readings can never occur with the overflow detection circuitry connected, because the instrument is locked off with a registration of zero (i.e., 00000) as soon as overflow occurs. Normal counting can only be resumed by pressing the reset button.

With the simple decoding and readout system shown in figure 1(a), the display follows the course of counting and only remains static once counting has stopped. In many applications this is not a disadvantage, but where counting samples must be taken at a rapid rate it may make reading rather difficult.

Because of this the more elaborate counter-based instruments often employ a more complex readout system known as **storage-type display.** A system of this type is shown in figure 1(b).

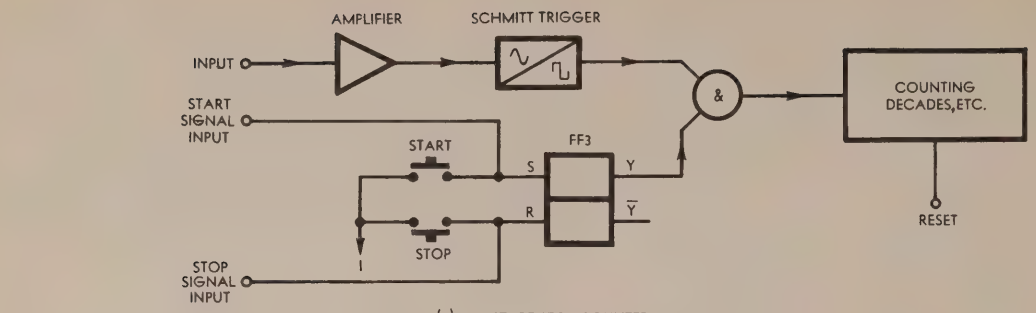
As may be seen, the system involves a complete second set of bi-stable elements, which act purely as storage registers to drive the decoding and readout circuit. The contents of the actual counting registers are transferred to the storage registers at the completion of each count, so that the display remains static during actual counting and changes—briefly—only after a new count is completed. Transfer of the new count from the counting registers to the storage registers is performed by a series of AND gates, activated by a suitably timed "store count" pulse.

Fairly obviously the provision of storage-type display adds considerably to the complexity of the basic counter. Thus even with economical neon tube bi-stable elements used in the storage registers the cost tends to rise quite sharply.

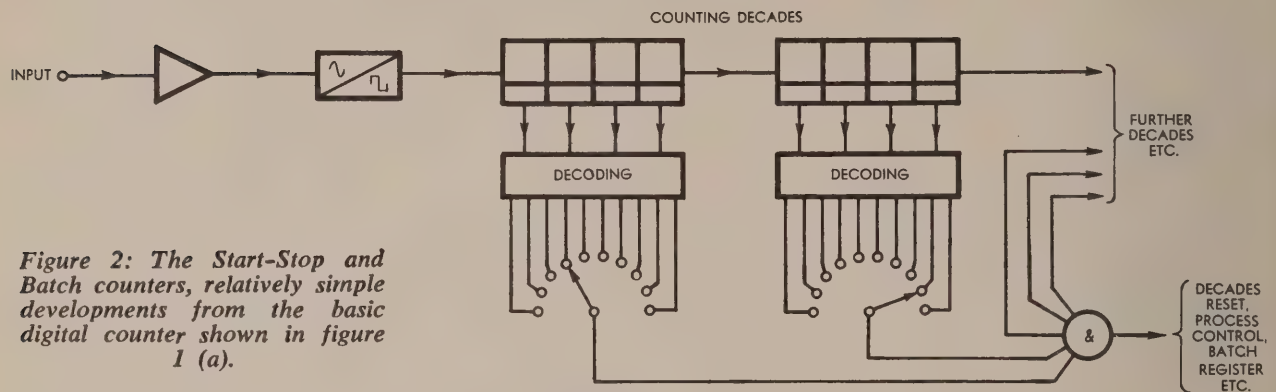
While the basic digital counter described thus far forms the basis of almost all digital measuring instruments, as noted earlier, it is almost never used as it stands. At the very least it is usually provided with some means whereby the input can be conveniently connected and disconnected, together with circuitry to ensure that the input signals to be counted are in suitably shaped digital form. Let us therefore pass on to a consideration of the ways in which the basic counter is extended and elaborated upon in practical instruments.

Perhaps the simplest practical counter is the **start-stop counter** illustrated by the logic diagram of figure 2(a). This is simply an instrument which counts and displays the number of pulses reaching it in a controlled interval of time. Instruments of this type may be used to count a wide variety of events—in fact, any type of event which may be transformed into an electrical pulse by means of a suitable transducer.

The start-stop counter typically adds to the basic counter of figure 1(a) a further AND gate, a gate-control flip-flop (FF3), circuitry to control the flip-flop, and input signal shaping circuitry.



(a) START-STOP COUNTER



(b) BATCH COUNTER

Figure 2: The Start-Stop and Batch counters, relatively simple developments from the basic digital counter shown in figure 1 (a).

The last-named usually consists of an amplifier and Schmitt trigger, as shown.

As may be seen the input to the counter is via the AND gate, which is under the control of the flip-flop FF3; hence the counter can only operate when FF3 is in the set (Y=1) state. Operation of the counter is thus simply a matter of setting FF3 for "start" and resetting it for "stop."

If a simple R-S flip-flop is used for FF3 it may be set and reset using push-buttons or equivalent control signals, as shown. An alternative arrangement is to use a gated R-S or J-K flip-flop connected for complementary-mode operation, in which case a single push-button or control signal input is sufficient.

As the simplest type of gated counter, the start-stop counter may be used to illustrate a fundamental source of error in instruments of this type—the error often called "one-pulse counter ambiguity."

The source of this error is the fact that a counter input gate opens and closes asynchronously relative to the input signal. As a result the gate may open and close at any instant during or between pulses. This can make the actual number of pulses admitted at a particular sampling either **one more** or **one less** than the average number of pulses occurring in the time interval concerned, depending upon the actual timing of the gate opening and closure relative to the pulses.

As a result of the one-pulse ambiguity, the LSD of the counter registration can be in error by one count in either direction. Hence the fundamental accuracy of most digital instruments based upon the gated counter is usually quoted as ± 1 count in the LSD.

A development from the basic counter which differs somewhat from the start-stop counter is the **batch counter,** used mainly in industrial processes to count and deal with articles or events as groups or "batches" of predetermined size. The essentials of a batch counter are shown in figure 2 (b).

As may be seen the unit is similar to the basic counter, although over-ranging is rarely used and overflow detection is usually unnecessary. The main difference is that in place of—or in addition to—the readout system there is a series of 10-position switches connecting to the inputs of an AND gate.

A switch is provided for each counting decade, and each switch may be set so that any one of the 10 decimal output lines of its decade may be connected to the AND gate input concerned. Hence the switches may be set so that the AND gate is operated at any particular stage of the count: if batches of 144 articles are to be handled the switches would be set to 144, and so on.

The output of the AND gate is used to activate the various process controls concerned with batch handling; it is also used to reset the counting decades so that the latter can commence counting the next batch. In addition it is normally used to operate a further counter registering the total number of batches handled.

Perhaps one of the most important configurations developed from the basic digital counter is that which effectively combines the start-stop counter with the double-pulse synchroniser discussed in last month's article. This is the arrangement shown in figure 3 (a), which can be used for measurement of time period, multiple period averages, frequency, and frequency ratios.

As with the start-stop counter the input to the counting decades is controlled by an AND gate; in this case it is usually known as the "Main Gate." The signals which it controls are those applied to "Input A" after these have received the necessary amplification and shaping.

The main gate is controlled by flip-flop FF1, which is controlled in turn by signals applied to its T terminal by a double-pulse synchroniser consisting of a second AND gate ("Control Gate") and another two flip-flops FF2 and FF3. The gating signals controlled by the synchroniser are derived from signals applied to the second main input terminal, "Input B."

Reset signals for the complete system are usually derived not from a push-button (although this is quite feasible), but from an adjustable low - frequency pulse generator usually called the **"Sample Generator."** The purpose of this generator is to periodically "wipe the slate clean," forcing the system to begin a fresh cycle of events and take a new sampling of the measurement concerned.

As each operating cycle commences with an overall system reset pulse from the sample generator, all three input control flip-flops are initially reset (i.e., Y= 0). Hence both main and control gates are initially closed.

The interconnection between the output terminals of FF3 and the gating terminals of FF2 are **logically** reversed. Thus with FF3 reset FF2 is in a position to be set to the Y=1 state by the next pulse to arrive at its T input terminal from the input B shaping circuitry. The next pulse applied to the input B terminal following the overall system reset therefore sets FF2, which in turn opens the control gate.

The **second** pulse to arrive from input B following the overall reset therefore finds the control gate open, and hence passes through to FF1. The latter is thus set to Y=1, opening the main gate and admitting pulses from input A to the counting decades.

Counting continues until the arrival of the **third** pulse from input B, which resets FF1 and re-closes the main gate. And in resetting, FF1 feeds a "lock-off" pulse back to the T terminal of FF3, to set the latter to Y=1.

The setting of FF3 prevents FF1 from being switched to the set state by further pulses reaching its T terminal, by

effectively applying a logical 0 to its J input gating terminal (logical 1 at Y of FF3 is logical 0 to J of FF1, due to the reversal of logic convention). The main gate is therefore prevented from re-opening on the fourth pulse to arrive from input B, and in fact remains closed until the next sampling cycle begins.

A second consequence of the setting of FF3 is that the input gating terminals of FF2 are biased such that the fourth pulse from input B resets FF2 and re-closes the control gate. Hence all activity of the input gating circuitry ceases after the arrival of the fourth pulse from input B, to recommence a new sampling cycle only when the sample generator initiates the next system reset.

The nett result of the foregoing cycle of events is that, following each reset pulse from the sample generator, the main gate of the counter is opened to admit signals from input A to the counting decades for an interval of time exactly equal to the time between two successive pulse transitions derived from input B.

Note that despite the precision with which the gate is opened and closed in synchronism with the input B signal, it still operates asynchronously relative to the input A signal. Hence the one-pulse ambiguity error noted in the case of the start-stop counter still applies. In fact, as already noted, it applies to all instruments based on the gated counter configuration.

There are four important measuring applications of the logical configuration of figure 3 (a), as previously mentioned, and these will now be discussed briefly in turn.

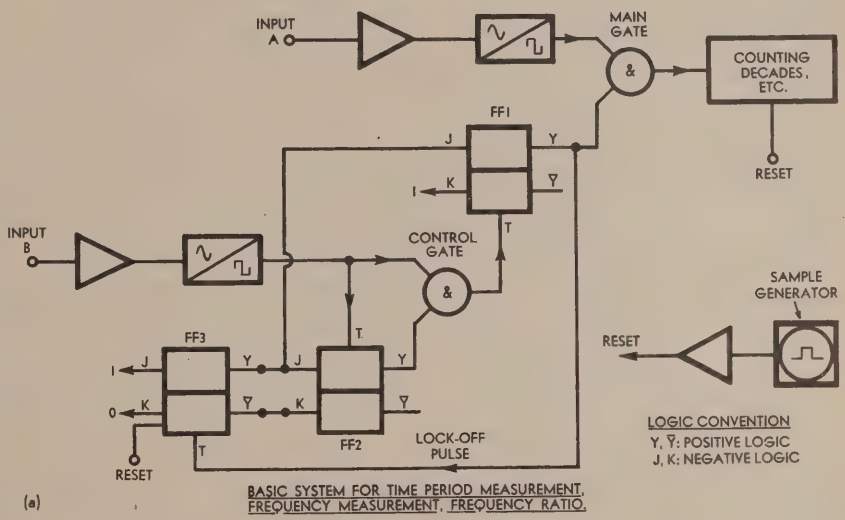
Time Period Measurements may be made as shown in figure 3 (b), by applying the signal to be measured to input B while a generator of accurately timed clock pulses is connected to input A. The main gate will then be opened to admit the clock pulses for a period equal to one cycle of the signal, so that the counter registration will be a measure of the period of the input signal in terms of the clock pulse period. If the clock pulse rate is 1MHz, for example, the counter will show the signal period in microseconds.

Multiple Period Average measurements are desirable where the period of the signal to be measured is short compared with that of the clock pulses available. In such cases the one-pulse counter ambiguity will represent an appreciable error in a single-period measurement.

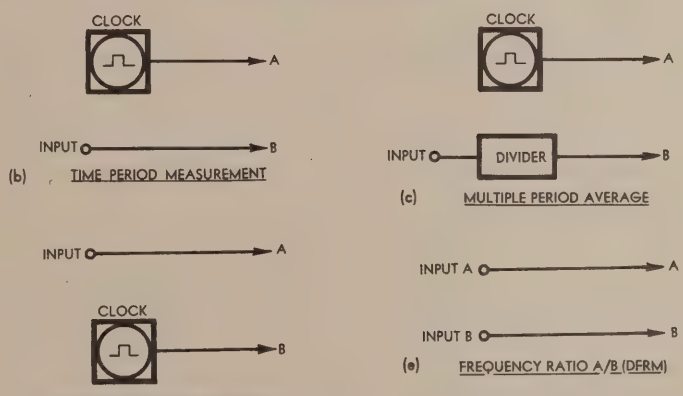
Figure 3 (c) shows the setup required for multiple period averaging. As may be seen it involves nothing more than the addition of a frequency divider in series with the input signal, the frequency division ratio of the divider determining the number of signal cycles averaged. Thus with a 10-times divider, the counter reading will represent the sum of 10 successive signal cycles; dividing the reading by 10 will thus give the signal period averaged over the ten cycles concerned.

Frequency Measurements may be made using the connections shown in figure 3(d). Here the input signal and clock are simply reversed from the positions used in time period measurement, so that the main gate is opened for a precise interval determined by the clock pulse period and the counter registers the number of input cycles arriving during that interval. Thus if the clock pulse rate is 1KHz and 700 pulses enter the counter the signal frequency must be (700 ± 1) KHz, allowing for the one-pulse ambiguity error.

The Hewlett - Packard 5216A counter - frequency meter, a compact general purpose instrument of moderate cost. (Picture courtesy Hewlett-Packard Australia Pty. Ltd.)



(a) BASIC SYSTEM FOR TIME PERIOD MEASUREMENT, FREQUENCY MEASUREMENT, FREQUENCY RATIO.



(b) TIME PERIOD MEASUREMENT

(c) MULTIPLE PERIOD AVERAGE

(d) FREQUENCY MEASUREMENTS (DFM)

(e) FREQUENCY RATIO A/B (DFRM)

Fig. 3

Frequency Ratio Measurements may be made using the arrangement of figure 3(e). Here the two signals to be compared are simply fed directly to the two inputs, so that signal B controls the gate admitting signal A to the counter. The counter will then read the number of cycles of input signal A occurring during one cycle of input signal B, which reading corresponds directly to the frequency ratio A/B. Thus if signal B is 35.7KHz and the reading of the counter is 100, the frequency of signal A must be (3570 ± 35.7)KHz, allowing once again for the one-pulse ambiguity error in the (ratio) reading.

Although the four applications of the configuration of figure 3(a) have been discussed separately, they are normally combined in a single multi-function instrument — the **"Digital Counter/Frequency Meter"** (DFM). A typical high-quality instrument of this type is pictured.

From the foregoing discussion it should be fairly apparent that a practical DFM will require not one, but a series of clock signals, in order to provide a number of measuring ranges for each mode of operation. Usually the required "family" of timebase signals are generated using a precision quartz oscillator together with a chain of frequency dividers.

In the more expensive instruments the readout system is arranged so that range and mode switching automatically indicate the appropriate units of measurement and position a decimal point on

the registration. Thus the display becomes quite unambiguous — for example "734.210KHz" — and leaves virtually no opportunity for reading error.

The maximum frequency which may be measured by a DFM is determined both by the maximum reliable counting speed of the initial (LSD) counting decade and by the maximum speed at which the input gating circuitry will operate with the required precision. These two factors limit the maximum frequency of direct measurement by present-day instruments to about 100-MHz.

While this is adequate for a great many purposes, there is a growing number of applications which require the accuracy of digital measurement at far higher frequencies. Thus various methods have been evolved to extend the basic measuring range without sacrificing accuracy. Three main methods have been evolved and these are shown diagrammatically in figure 4.

Probably the simplest of the three methods is **prescaling,** shown in figure 4(a). As the name suggests this simply involves a suitable high-speed scaler/frequency divider connected ahead of the DFM itself. Usually the prescaler divides by a multiple of 10, to allow convenient mental multiplication of the DFM reading when calculating the actual input frequency. Thus if the DFM reads "83.015MHz" with a x10 prescaler in use, the input frequency would clearly be 830.15MHz.

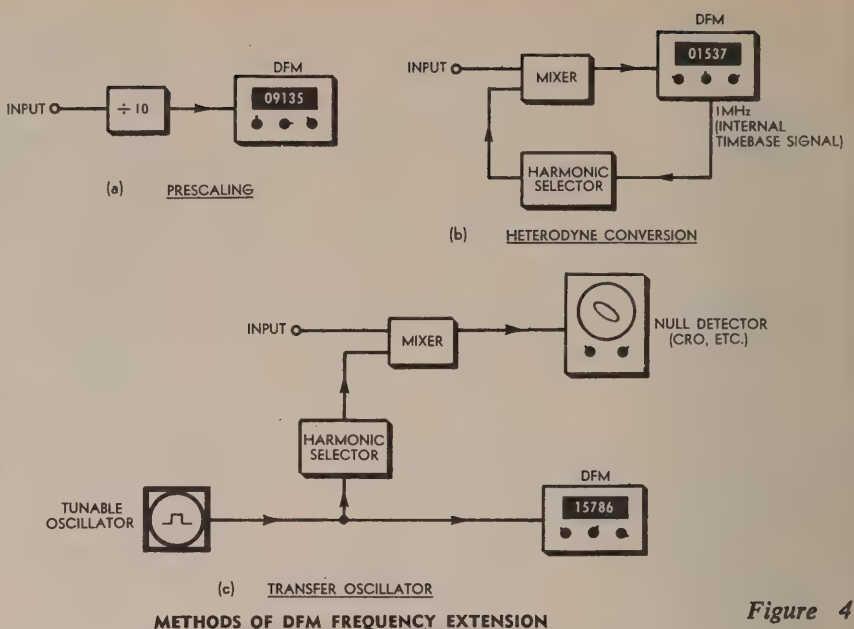
While prescaling does not affect the percentage accuracy of the DFM, it does degrade the absolute resolution. This can be seen in the above example, where the significance of the LSD is 1KHz (.001MHz) for the basic DFM but only 10KHz with the prescaler connected. The percentage accuracy would here remain unchanged at ± .001 p.c., which corresponds to the 1-pulse ambiguity in the basic count (neglecting other sources of error such as timebase drift, etc., which are normally somewhat smaller than .001 p.c.).

At present the frequency extension possible by prescaling is limited to about 300MHz, because few devices and circuitry currently available are capable of frequency division at higher than this figure.

Extension of the measuring range further than 300MHz usually involves the second method, **heterodyne conversion.** This is illustrated in figure 4 (b). Here the internal timebase clock of the DFM (usually 1MHz or 10MHz) is multiplied up so that selected harmonics can be heterodyned with the input signals in an RF mixer. The resultant beat frequency is then measured by the DFM in the usual way.

The actual input frequency is found by adding or substracting the DFM reading from the known frequency of the timebase harmonic in use. To find out which of the two operations is appropriate it is usually necessary to obtain readings with a number of adjacent timebase harmonics in order to determine the two harmonics between which the signal is located.

Although care must be taken with a heterodyne conversion setup to ensure that the correct measurement is taken, if this is done the accuracy and resolution can both be better than with prescaling—despite the fact that conversion measurements may be made up to 3GHz (3000MHz).



METHODS OF DFM FREQUENCY EXTENSION

Figure 4

The resolution is basically the same as that of the basic DFM, as the DFM itself reads the absolute difference between the input signal and the selected timebase harmonic. The percentage accuracy is somewhat more complex, being determined by the timebase drift and the timebase harmonic in use as well as the 1-pulse counter ambiguity, but with careful design it can be made better than that of the basic instrument.

Where measurements must be made at frequencies even higher than 3GHz it becomes necessary to use the third method of range extension, the **transfer oscillator method.** This is illustrated in figure 4 (c).

Basically the idea is that a highly stable tunable oscillator is used as an intermediate transfer device between the input signal and the DFM. A selected harmonic of the oscillator is nulled to zero beat against the input, while the DFM actually reads the frequency of the transfer oscillator; the input frequency is then found by multiplying the DFM reading by the order of the harmonic in use.

Thus if the DFM shows the transfer oscillator to be 57.9702MHz at zero beat using the 110th harmonic, the input signal frequency would be (57.9702 x 110) or 6376.72MHz (6.37672GHz).

Although transfer oscillator measurements can be made up to about 20GHz using present-day equipment, operation is rather more tricky than with the other two methods. It is easy to use a false null, particularly if the signal is noisy; it is also somewhat difficult on occasion to determine which transfer oscillator harmonic is actually involved. Apart from these problems there is the risk of an error in calculations because the operator must multiply a many-digit DFM reading by a harmonic number which may lie anywhere between 1 and 99 or more.

Let us now consider a further type of digital measuring instrument — the **Digital Voltmeter** or "DVM," used as the name would suggest to measure the amplitude of signal voltages.

There are three main types of DVM, usually referred to as the Ramp, Integrating and Potentiometric types respectively. The basic logic arrangement for each type is shown in figure 5. All three types are basically DC measuring systems and must be preceeded by rectifier and filter circuitry for AC measurements.

The **Ramp type DVM** consists of a gated counter connected to a clock oscillator, with the gate controlled by an analog comparator. The positive comparator input connects to the voltage to be measured, while the negative input connects to a linear analog voltage ramp generator which is enabled by a Schmitt trigger whenever an input signal is applied.

When an input is connected, the comparator opens the counter gate and clock pulses enter the counter. At the same time the ramp generator is enabled, and the voltage at the comparator negative input rises linearly to approach the input voltage. As soon as the ramp voltage equals the input voltage, the comparator closes the counter gate and counting stops. The counter registration is then directly proportional to the input voltage.

If F is the clock frequency, R the slope of the voltage ramp, and Ex the voltage being measured, the counter reading N is given by

$$N = Ex.F/R$$

Thus if the ramp generator produces a voltage ramp of 50 volts per second and the clock frequency is 50KHz, a voltage of 4.725 volts will produce a reading of "4725," to which the appropriate decimal point is easily added.

The basic limitation on the sensitivity and resolution of a ramp-type DVM is the "aperture" of the analog comparator. It may be recalled that this term refers to the magnitude of the input voltage difference required by the comparator for its output to change value. Naturally the smaller the voltage difference to which the comparator will respond, the greater the instrument sensitivity and the smaller the increment in input voltage which it will recognise or "resolve."

The aperture of the comparator together with the linearity of the ramp generator and the frequency stability of the clock also determines the basic accuracy of the instrument. As a result of the problems associated with these factors it is difficult to construct a ramp-type DVM with an accuracy of better than (±.05 p.c. of full scale ±1 count in the LSD).

The ramp-type instrument is quite susceptible to noise and AC superimposed upon the input voltage, as such signals effectively change the input amplitude at the comparator and hence cause counting to be stopped erratically and indeterminately. It is therefore necessary to provide such instruments with fairly extensive low-pass input filtering if measurements are to be unaffected by superimposed noise and AC.

The reading rate of ramp-type DVMs therefore tends to be rather slow, both because of the time necessary to generate the ramp itself and because of the long time-constant of the required input filter.

As a result of its rather numerous shortcomings, the ramp-type DVM tends to be used mainly as an "economy" class instrument.

The **Integrating DVM** consists basically of a voltage-to-frequency converter connected to a digital frequency meter, as shown in figure 5(b). The converter produces a series of pulses whose frequency is proportional to the instantaneous input voltage; the frequency of the pulses is then measured by the DFM to produce the final reading.

Because the DFM reading is simply a sum or integration of the V-F converter output pulses during the DFM gating time, it represents the **average** converter frequency during that time. Hence the integrating DVM is an average reading instrument and will ignore quite large amounts of AC superimposed upon the DC voltage being measured, provided that an integral number of cycles of the AC signal occur during the gating period.

As the most usual source of AC superimposed on DC signals is the power mains, it is usual with integrating DVMs to make the gating time either equal to or a submultiple of the period of the mains frequency. This gives the instrument the ability of measure quite small DC voltages while ignoring large amounts of superimposed AC.

A simple integrating instrument of the type shown is thus able to measure DC signals with virtually full accuracy even in the presence of a superimposed AC signal of identical peak amplitude. If a more elaborate system with a bi-directional counter is used, this performance can be improved upon still further —even to the point where the superimposed AC is 30 to 40 times the amplitude of the DC voltage being measured. Usually the only factor governing the level of superimposed noise is overload of the V-F. converter.

The accuracy and linearity of the integrating DVM depend mainly upon the linearity of the V-F converter, as one might expect. To attain high levels of accuracy the converter design is therefore quite complex, and the cost of the instrument tends to be relatively high as a result. The accuracy of a typical high-quality integrating DVM of modern design is (±.01 p.c. of reading ±.005 p.c. of full scale ± 1 count in the LSD).

The sensitivity and resolution of the integrating DVM depend mainly upon the conversion factor of the V-F converter. The higher the converter output frequency produced for a given voltage input, the greater the instrument sensitivity. However it is very difficult to maintain converter linearity over a useful input range if the conversion factor is increased much above 500KHz/V.

The third main type of digital voltmeter is the **Potentiometric DVM,** illustrated in figure 5 (c). As may be seen this type is actually an analog-to-digital converter of the type discussed in the eighth article in this series.

As with the ramp type DVM the potentiometric type uses an analog comparator to compare and match the input voltage with an internally generated "bucking" voltage. However, in contrast with the ramp type instrument the matching voltage is produced in this case by a digital-to-analog converter (DAC) connected, in parallel with decoding and readout circuitry, to the output of a counter or similar logical sequencer.

Operation is somewhat similar to that of the ramp type instrument. When input is applied, the comparator opens the gate and pulses enter the counter or sequencer. The latter then proceeds to generate digital output signals, which are converted into equivalent analog voltages by the DAC. As soon as the DAC output voltage equals the input voltage, the comparator immediately closes the gate and operation ceases until the next sample is required—with the readout system displaying the digital equivalent of the input voltage.

If a simple counter is used as the logical sequencer, operation tends to be rather slow because the matching voltage must rise from zero as a "linear" staircase signal for each sample measurement. Hence it is more common to employ a logical sequencer other than a simple counter, as operating speed is usually fairly important.

Probably the most usual logical sequencer employed is the so-called "sequential approximation" type, which uses a shift register and decoder to add or subtract decreasingly small amounts to the DAC input.

As soon as input is applied, the sequencer immediately generates a digital figure of **half** the range full-scale value. If the DAC output then exceeds the input, it **substracts** a figure of one-quarter full scale; on the other hand if the input is still larger than the DAC output it **adds** a figure of one-quarter full scale. Then depending again on whether the DAC output is more or less than the input, it subtracts or adds a figure of one-eighth full scale, and so on. Thus the instrument rapidly "zeroes-in" on the input voltage by a series of successive and decreasing approximations.

The sensitivity and resolution of the potentiometric DVM are limited mainly by the aperture of the analog comparator, as with the ramp type instrument.

Accuracy is fairly easy to achieve with the potentiometric DVM because the voltage matched against the input is generated using a precision voltage supply and precision resistor networks in the DAC—there are virtually no problems similar to those of generating a linear analog voltage ramp or achieving linear voltage-to-frequency conversion. Accuracy is typically about (±.005 p.c. of full scale ± 1 count in the LSD).

However, unlike the integrating DVM —and like the ramp type DVM—the potentiometric type has no inherent noise rejection. Therefore a considerable amount of input filtering is again necessary if the measurements are to be unaffected by superimposed AC and noise.

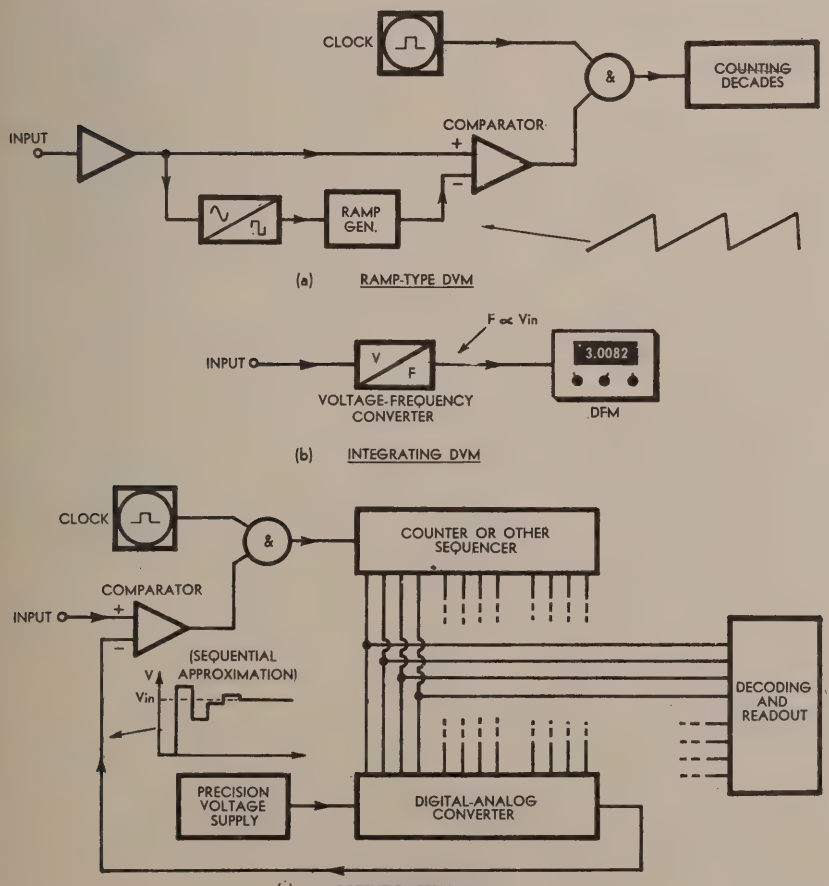


Fig. 5 **DIGITAL VOLTMETERS**

The filter time constant tends to make the reading speed quite slow, even for instruments with a sequential approximation or similar sequencer in place of a counter. In terms of noise immunity and reading rate the potentiometric DVM thus tends to be very similar to the ramp type instrument.

By combining the integrating and potentiometric techniques it is possible to produce a DVM which possesses many of the advantages of both types, with few of the disadvantages. The instrument pictured is of this type, offering 1 microvolt resolution together with an accuracy of (±.005 p.c. of reading or ±.0005 p.c. of full scale, whichever is greater).

To conclude the present survey of digital instruments let us now consider two instruments somewhat less common than those already discussed. These are the digital phasemeter and the digital thermometer.

"Analog" measurements of phase differences between AC signals are usually made using what are already basically digital techniques, as figure 6(a) shows. The two signals to be compared are first amplified, and then squared using Schmitt triggers. Following the triggers one signal is inverted and the two are then fed to an AND gate.

The output of the AND gate is a series of constant-amplitude pulses, whose period is equal to that of the input signals (1/frequency), and whose width is determined by the measured phase difference. In fact the width/period ratio of the pulses will be exactly equal to the measured phase difference expressed as a fraction of a full cycle (360 degrees), irrespective of the frequency of the signals being compared.

As a result of this identity an average-reading analog meter movement fed with the gate output pulses gives a reading directly proportional to the phase difference, and can be calibrated directly in degrees (or radians, if required).

The maximum reading produced on the meter will correspond to a phase difference of 180 degrees, as the AND gate deals only with half-wave pulses. For measurement of phase differences from 180-360 degrees, the inverter may be switched out of the B signal line. This virtually subtracts 180 degrees from the B signal, to bring the phase difference within the 0-180 degree measuring range.

The Digital Phasemeter uses the same basic phase measuring circuitry as in figure 6(a), but in place of the analog meter it uses once again a variation of the now-familiar gated counter. The usual logic configuration is shown in figure 6(b).

A clock oscillator is used to supply input pulses to the gated counter, with the phase measuring circuit operating the counter gate. The clock oscillator also has connected to it a second counter used to control the number of output pulses delivered during each measurement. As soon as the clock has delivered the appropriate number of pulses it is disabled by the counter, stopping operation until the operator initiates a fresh measurement by pressing the "sample" button.

The counter controlling the clock is set so that for each measurement the clock produces a number of pulses equal to an appropriate decimal multiple of 360; which decimal multiple is used is determined by the number of digits

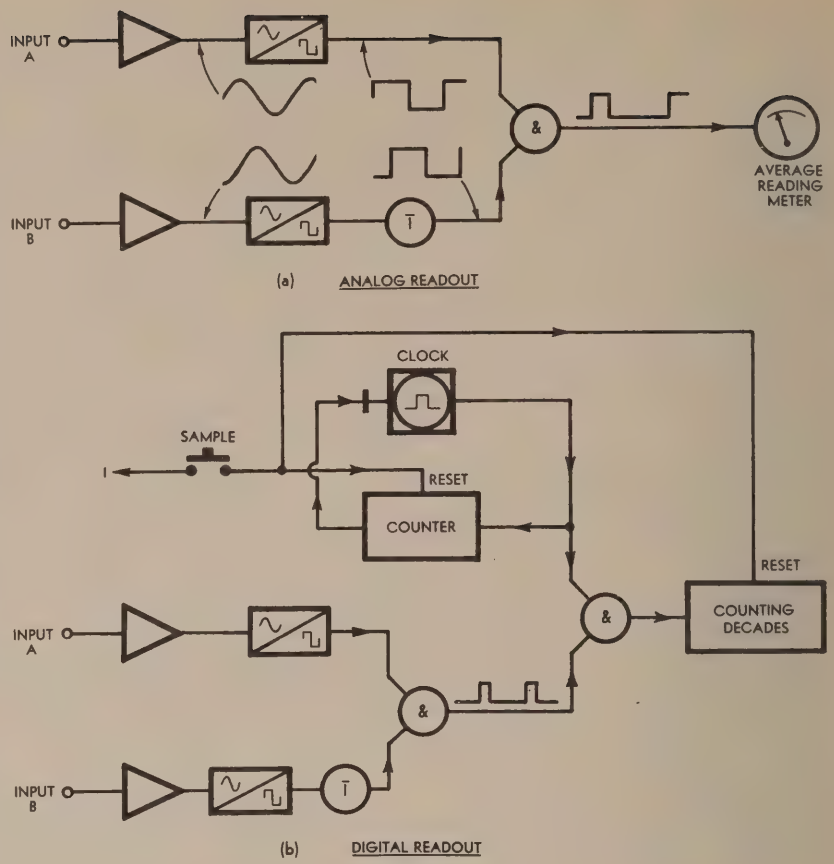


Fig. 6

PHASEMETERS

available on the main counter. Thus for a 5-digit instrument the clock counter would be arranged to permit the clock to deliver 36,000 pulses per measurement.

The pulses delivered by the phase measuring gate to the main counter gate have a width/period ratio directly proportional to the phase difference being measured, as we have seen. Hence **provided** that the clock oscillator frequency is very much greater than the frequency of the signals being compared, the proportion of the total clock pulses fed to the counter will be exactly equal to the same phase difference. This is because the counter gate will be open for exactly that proportion of the total time.

For a phase difference of 20 degrees, for example, the width/period ratio of the pulses produced by the phase measuring gate will be 20/360=1/18. The counter gate will therefore be open for this proportion of the total time. Thus if the total number of pulses supplied by the clock is 36,000 only 36,000/18= 2,000 will enter the counter. The counter will thus read "2000" immediately counting ceases.

Similarly a phase difference of 57 degrees will produce a reading of "5700," and so on. It should be fairly obvious that by fitting the counter with a suitable decimal point indicator, its reading may be read directly in degrees to two decimal places. The accuracy of measurement will depend mainly upon the ratio between the clock pulses and the frequency of the signals being compared.

As with the analog instrument the basic measuring range will be 0-180 degrees, and measurements of phase differences greater than 180 degrees are made by switching out the B signal inverter. It is possible to produce a full 0-360 degree measuring range, if necessary, by replacing the AND gate with a slightly more complex arrangement using a simple R-S flip-flop.

The Digital Thermometer, as its name suggests, is an instrument for making accurate measurements of absolute tem-

GLOSSARY OF IMPORTANT TERMS

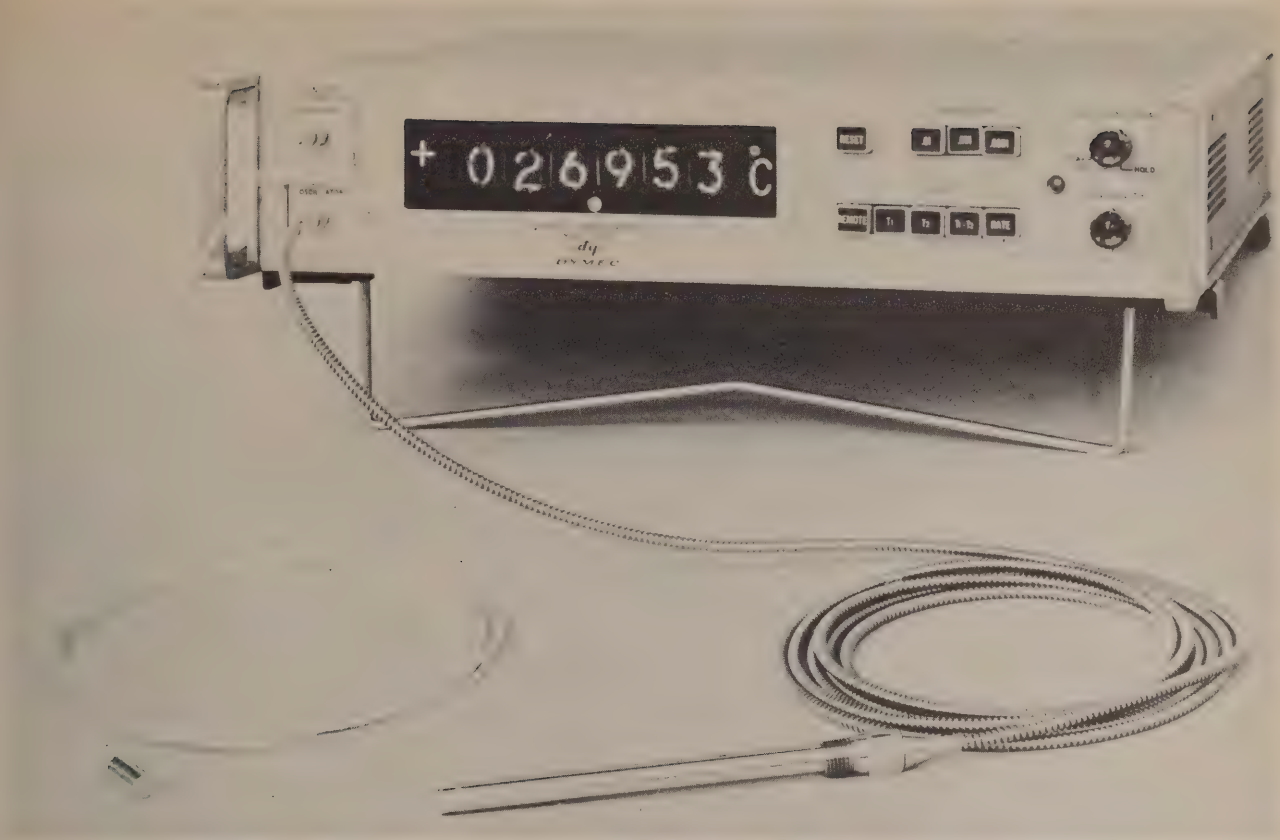
LSD: The least significant digit of a number or its representation.

MSD: The most significant digit of a number or its representation.

Overflow: Return of a counting register to "zero" registration upon the arrival of a pulse subsequent to that which "fills" it to capacity.

Over-ranging: Extension of the capacity of a counter at modest cost by means of partial-decade counting circuitry subsequent to the most significant full decade. Often comprises only a single flip-flop.

Storage-Type Display: Use of duplicate storage registers to store the result of each counting sample for convenient readout.



The Dymec Model 2801A, a digital thermometer capable of measuring temperature and temperature differences within .02 degree C. over the range 0-100 degrees. (Courtesy Hewlett-Packard Australia Pty. Ltd.)

perature and temperature differences. The high performance instrument of this type pictured has a range of (−40—+ 240 degrees C.), with a resolution as low as .0001 of a degree and an accuracy of .02 p.c. over the range 0-100 degrees C.

The basic logic configuration for an instrument of this type is shown in figure 7.

The heart of the unit is a temperature sensor consisting of a special quartz crystal element. In contrast with the crystals normally used in electronics, which are cut to give a temperature coefficient of frequency as near to zero as possible, the element is in this case cut with extreme precision to give a highly **linear** coefficient.

Thus when the crystal is connected into an oscillator circuit, the frequency of the oscillator will change linearly with temperature. In the instrument shown the resonant frequency of the crystal is 28.2MHz at 0 degrees C, and changes by (1000Hz ± 150Hz)/degree over the full range of (-40—+240 degrees C. Over the (0—+100) degrees range it is accurate to within 20Hz/degree.

The sensor and oscillator form a temperature-to-frequency converter; hence to obtain a digital reading it is simply necessary to perform a frequency measurement on the oscillator output. As may be seen this is done using a heterodyne conversion in order that the meter can give a zero reading at 0 degrees C.

The oscillator output is mixed with a 28.2MHz signal from a stabilised reference oscillator, so that at 0 degrees the mixer output frequency is zero. Then as the temperature deviates from 0 degrees, the mixer output frequency rises from zero by 1000Hz per degree. This frequency is then measured by the DFM using a timebase signal derived from the reference oscillator.

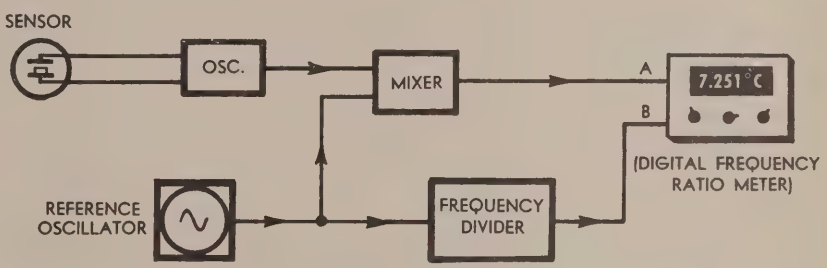


Fig. 7 **THE DIGITAL THERMOMETER**

Thus when the sensor temperature is +7.251 degrees, the mixer output frequency is 7,251Hz. Using a 1Hz timebase signal from the reference oscillator the DFM will thus produce a reading of "7251." With suitable circuitry to detect the direction in which the sensor frequency has changed and provision for display of a decimal point, the instrument thus reads directly in degrees.

For temperature differential measurements, two sensors are used; the output of the second sensor being fed to the mixer in place of the 28.2MHz reference signal. The timebase signal is derived from the reference oscillator as before. As the mixer output frequency will now represent the temperature difference between the two sensors at the rate of 1,000Hz/degree difference, the instrument will again read directly in degrees.

In closing the present chapter it is hoped that the foregoing discussion of the more common digital instruments will be found a worthwhile one, despite the necessary brevity. Further information on certain of the instruments discussed may be found from the works listed below in the bibliography.

SELECTED BIBLIOGRAPHY:

DEAN, K. J., **Digital Instruments,** 1965. Chapman and Hall Ltd., London.

KAY, B. G., "Selecting the Right Digital Voltmeter," in **Electronics,** V.39, No. 7 (April 4, 1966), p.84.

MILLMAN, J., and TAUB, H., **Pulse, Digital and Switching Waveforms,** 1965. McGraw-Hill Book Co., Inc., New York.

SELECTION GUIDE: "Which Electronic Counter?" in **International Electronics,** V.9, No. 6 (June, 1965), p.22.

SCHMID, H., "Digital Meters for Under $100," in **Electronics,** V.39, No. 24 (November 28, 1966), p.88.

Chapter 11

DIGITAL COMPUTERS

Digital arithmetic—binary addition and subtraction—half adders and full adders—serial, parallel and two-step parallel addition—special-purpose and general-purpose computers—programmed digital functions, and the stored program—the main sections of a general-purpose digital computer and their operation—basic concepts of programming.

In the preceding discussion of digital instruments we have seen that counters, timers, frequency meters, voltmeters and various other instruments can be produced by interconnecting flip-flops, logic gates and other digital elements. Let us now look at the ways in which these elements are used in modern general-purpose digital computers.

Right at the outset it should be stressed that modern digital computers are extremely complex machines, and that in the necessarily restricted space available here it will barely be possible to give a general view of the basic principles of computer operation. However, it is the author's hope that the discussion which follows will serve at least to give readers an insight into basic computer principles, sufficient both to prepare them for and to interest them in further detailed study.

As we have seen in earlier chapters, digital circuits deal with electrical signals which may be interpreted as representations of logical statements, and the statements concerned may often be numerical statements or numbers. We have also noted that Russell, Whitehead and others have shown that logic is in fact the foundation of mathematics, and that Shannon has demonstrated the importance of logic as a basis for digital circuit design. From these premises, it follows that, by interconnecting digital elements in suitable logical configurations, it proves possible to construct digital circuits capable of performing electrical operations corresponding to arithmetic operations upon numbers.

We have also seen in earlier chapters that electrical and electronic circuits are, by their nature, best suited for the representation of two-valued quantities and that, as a result, most modern digital circuitry handles numbers in either pure binary form or in one of the many BCD codes. Digital "arithmetic" circuitry is therefore almost completely concerned with performing binary or modified binary arithmetic.

The rules of binary arithmetic, although less familiar, are actually quite analogous with those of the decimal system. Not only this but they are actually simpler than their decimal counterparts. We have already seen in chapter 3 how simple the rules of addition become in binary notation, resolving into the simple table:

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 0 \text{ and 1 to carry} \quad \ldots\ldots (1)$$

Hence the binary addition of 1101 and 110, equivalent to the decimal addition of 13 and 6, looks like this:

$$\begin{array}{r} 1101 \\ 110 \\ \hline 10011 \end{array}$$

Examination of this simple example will show the similarity between decimal and binary addition. And if the reader checks "10011" he will find it to be the binary equivalent to decimal 19.

The rules for binary subtraction are equally simple as those for addition:

$$0 - 0 = 0$$
$$1 - 0 = 1$$
$$0 - 1 = 1 \text{ but 1 borrowed}$$
$$1 - 1 = 0 \quad \ldots\ldots (2)$$

As an example consider the following, which shows the subtraction of 101 (decimal 5) from 10110 (decimal 22):

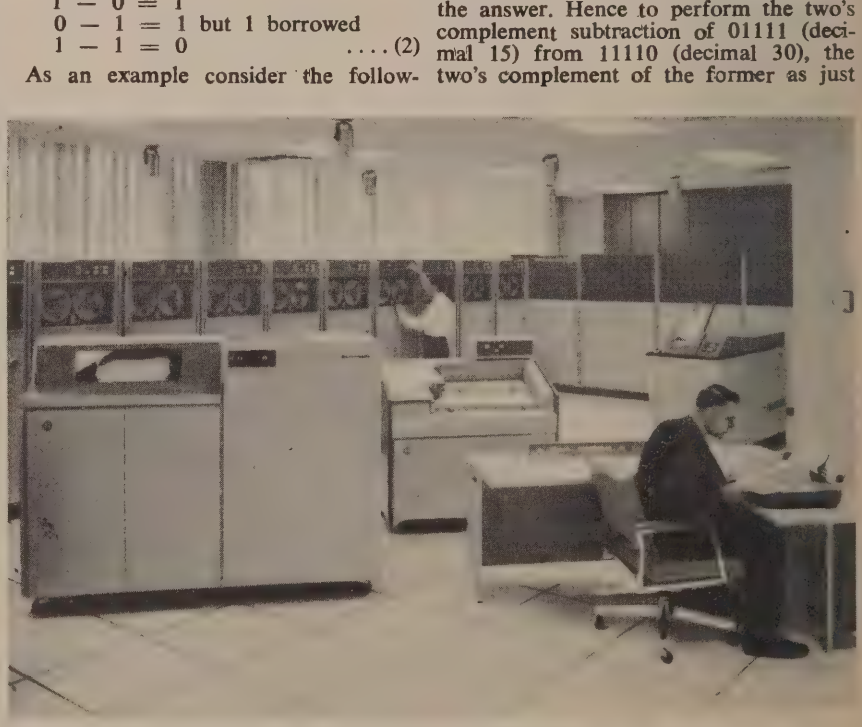
$$\begin{array}{r} 10110 \\ 101 \\ \hline 10001 \end{array}$$

Although it is quite feasible to perform binary subtraction in this manner, it is often found more convenient to use an alternative method known as complementary addition. This consists of forming the complement or negative of the subtrahend, which is then added to the diminuend using a slightly modified binary addition.

A popular way of performing subtraction by complementary addition is by using the so-called "two's complement" method. The **two's complement** of a binary number is formed by individually complementing each bit of the number and finally adding one. An example should show what this involves:

$$\text{number} = 01111 \text{ (decimal 15)}$$
$$\text{2's complement} = 10000 + 1$$
$$= 10001$$

In the addition phase of two's complement subtraction the only difference from "normal" binary addition is that any carry-over having a greater significance than that of the most significant bit of the diminuend is ignored — at least, with regard to the **magnitude** of the answer. Hence to perform the two's complement subtraction of 01111 (decimal 15) from 11110 (decimal 30), the two's complement of the former as just



A large general-purpose computer system. At rear are magnetic tape units, with a high-speed printer in the left foreground. (Courtesy Burroughs Corporation.)

derived is added to the latter, ignoring the carry-over:

$$\begin{array}{r} 11110 \\ 10001 \\ \hline (1)01111 \end{array}$$

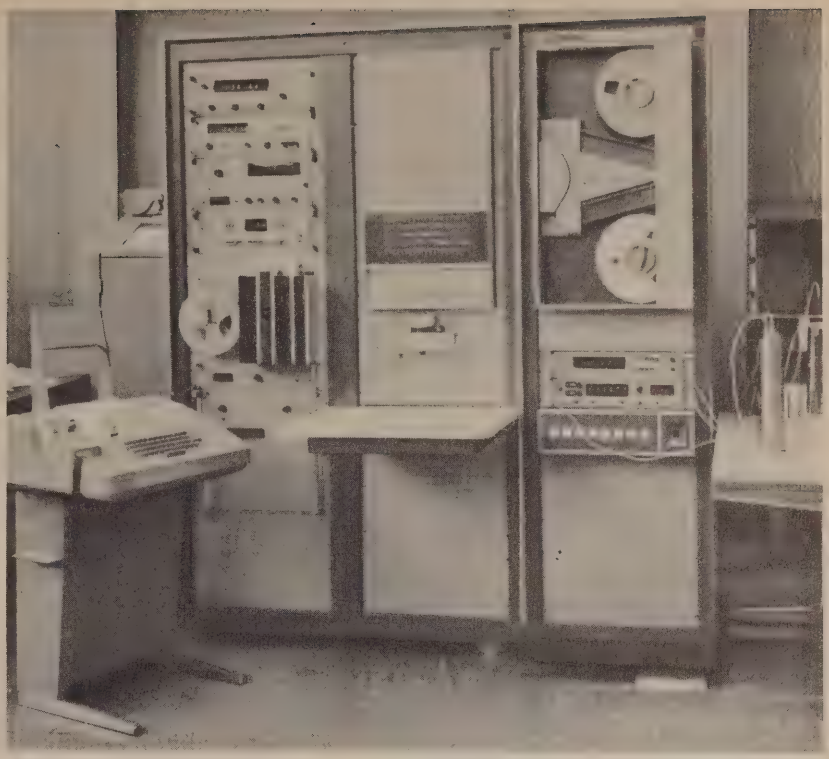
Here the answer is regarded as 01111, giving the correct answer equivalent to decimal 15.

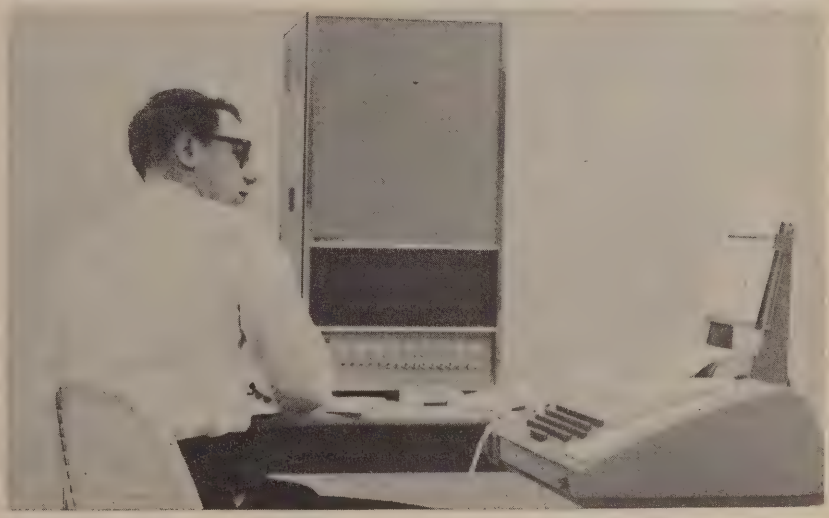
It may be seen that binary addition and subtraction may both be performed by a circuit configuration capable of performing binary addition. In view of this it should be worthwhile at this point to look briefly at representative examples of the many logic circuit configurations which have been developed to perform the function of binary addition. We will consider first the circuitry required for addition of corresponding bits, and then the ways in which this basic adding circuitry is used to perform addition of whole numbers.

Inspection of the elementary rules of binary addition given in expressions (1) should show that simple binary addition is equivalent to a complex logical operation involving two inputs ("addend" and "augend") and two outputs. One output, the "sum," must be true (1) only when one and only one of the inputs is true; while the second or "carry" output must be true only when both inputs are true. Thus simple binary addition may be regarded as equivalent to the logical operation defined by the truth table:

INPUTS		OUTPUTS	
ADDEND A	AUGEND B	SUM S	CARRY C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

It may be seen that the "sum" output consists of an Exclusive - OR function of the two inputs, while the "carry" out-





The HP-2116A, a compact high speed general-purpose computer, particularly suitable for interconnection with other digital equipment. At top it is shown integrated into a typical research instrumentation system, while below is the basic setup for "straight" computation. (Courtesy Hewlett-Packard Australia Pty. Ltd.)

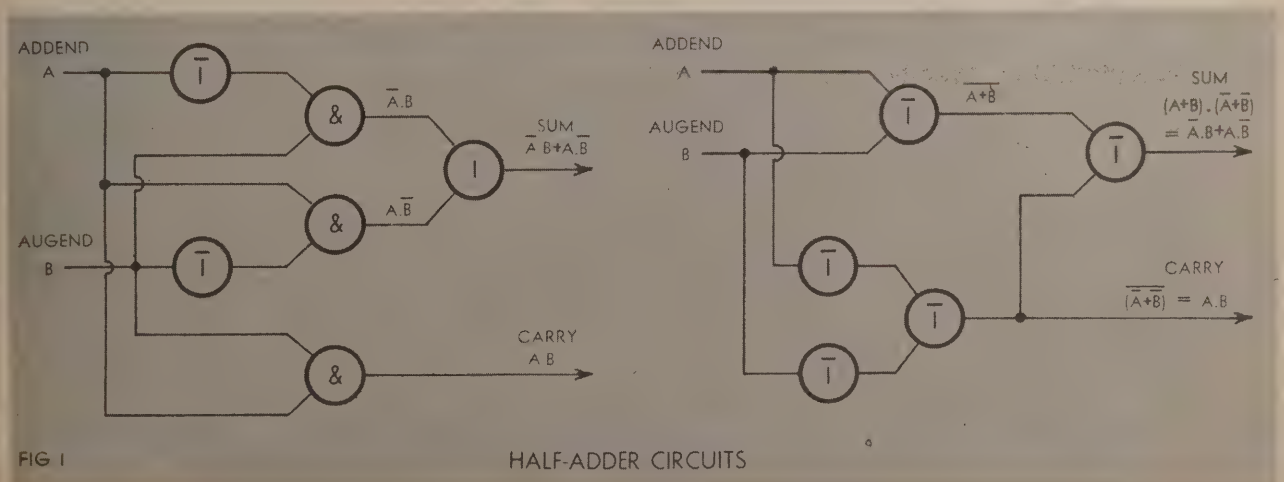


FIG 1 HALF-ADDER CIRCUITS

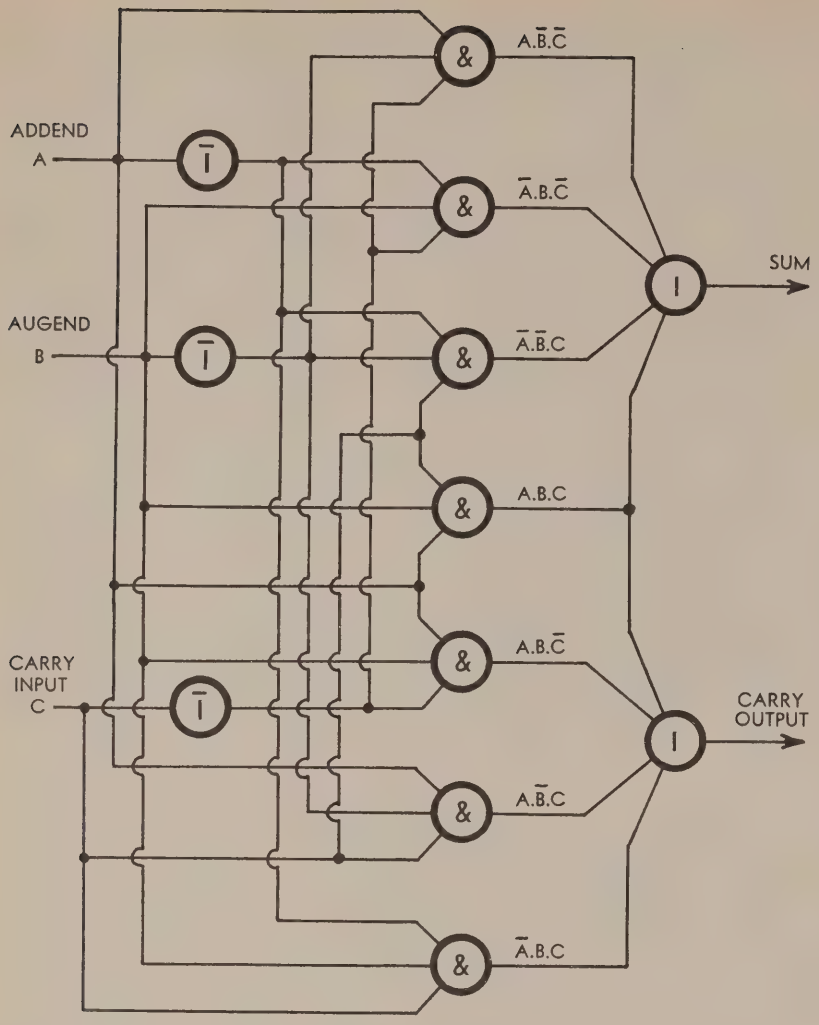


FIG 2 A FULL BINARY ADDER

put consists of an elementary AND function. Typical logic element configurations used to realise these functions are shown in figure 1; in each case the input inverters are not required if the complements of the input bits are already available.

Circuits of the type shown in figure 1 are normally termed **"half-adders,"** because generally they provide only half the circuitry required for practical binary addition. The reason for this is that it is rare for two bits to be added together in isolation — normally there will be a third bit to be taken into account, representing the carryover from the addition of the bits ranking next below those considered.

A full binary adder must, in fact, perform the operations defined by the following truth table:

INPUTS			OUTPUTS	
CARRY INPUT C	ADDEND A	AUGEND B	SUM S	CARRY C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

There are many ways of producing a full adder, one practical configuration being shown in figure 2. Another approach is to combine two half-adders, with an OR gate to produce the final carry output.

Let us now look at the ways in which full-adders or combinations of half-adders and other elements are used in performing addition of multi-bit binary numbers.

In general there are two basic methods used in handling binary numbers in digital circuitry, termed respectively the **serial** method and the **parallel** method. Numbers are handled sequentially in bit-by-bit fashion (usually LSB first) with the serial method, whereas with the parallel method they are handled "in toto" by treating all bits simultaneously. Each of the methods has advantages: the serial method generally offers circuit simplicity and economy, while the parallel method is generally capable of operation at considerably higher speeds.

Addition of numbers using the serial method typically involves a single full-adder unit, into which the numbers to be added together are fed bit-by-bit — corresponding bits at a time, starting from the LSBs—by means of shift registers. The carry-over from each bit pair to the next-most-significant pair is usually stored between additions in an auxiliary flip-flop. The sum may be stored in a third shift register, or alternatively it may be fed back to one of the registers initially containing the addend or the augend.

Figure 3 shows a serial adder for numbers of up to six bits, in which the sum is returned to the addend register. It operates as follows: initially, both registers and the carry flip-flop are cleared by a pulse applied to a common reset line (not shown). Then the addend and augend numbers are entered into their corresponding registers. This may be done serially, by shifting the numbers into the registers via the serial inputs shown (this will involve six clock pulses) or in parallel fashion by means of additional set gates individually connected to the register flip-flops. When the two numbers are entered into their respective registers the application of a sequence of six clock pulses will cause their addition to take place.

As the A and B inputs of the adder connect to the outputs of flip-flops 6 and 12, prior to the arrival of the first clock pulse the adder inputs will correspond to the LSB of each number. Hence the adder will produce the appropriate sum and carry signals, which will be connected to the gating inputs of flip-flop FF1 and the carry flip-flop respectively. Hence upon the arrival of the first clock pulse, these flip-flops will effectively store the LSB sum and carry; at the same time the initial contents of both the addend and augend registers will be shifted one place to the right.

Following the first clock pulse, therefore, FF1 will contain the LSB of the sum; FF2 and FF8 will contain the MSBs of the original numbers which were initially in FF1 and FF7; FF6 and FF12 will contain the next-least significant bits of the numbers; and the carry flip-flop will contain the LSB carry. The outputs of the adder will accordingly correspond to the sum and carry of the next-least-significant bits.

The arrival of the second clock pulse will cause this sum and carry to be entered into FF1 and the carry flip-flop, as before. It will also shift the LSB of the sum into FF2, and the remaining bits of the addend and augend into FF3-FF6 and FF9-FF12. And so on for the remaining clock pulses. It may be seen that following the sixth clock pulse, FF1-FF6 of the addend register will contain the total sum, FF7-FF12 of the augend register will be cleared, and the carry flip-flop will contain any carry-over from the MSB addition.

Serial addition of large numbers tends to be prohibitively time-consuming because each bit pair and the appropriate carry must be added, in turn, with the initiating clock pulses spaced to allow for element switching times and wiring delays. In contrast, the parallel method of addition is considerably faster, since all additions are performed simultaneously. However completely parallel single-step addition requires a complete set of full adders — one for each bit position. This tends to make such an adder extremely costly.

Consequently, where extreme speed is not required, computer manufacturers frequently employ a compromise technique known as "two-step" parallel addition. This permits a significant reduction in the number of circuit elements required, yet allows addition to be performed in a time interval only twice that required for single-step addition.

Two-step parallel addition involves separation of the addition process into two consecutive operations: **half-add,** in which the Exclusive-ORs of all the input bit pairs are produced, and **carry,** in which are produced and added in all

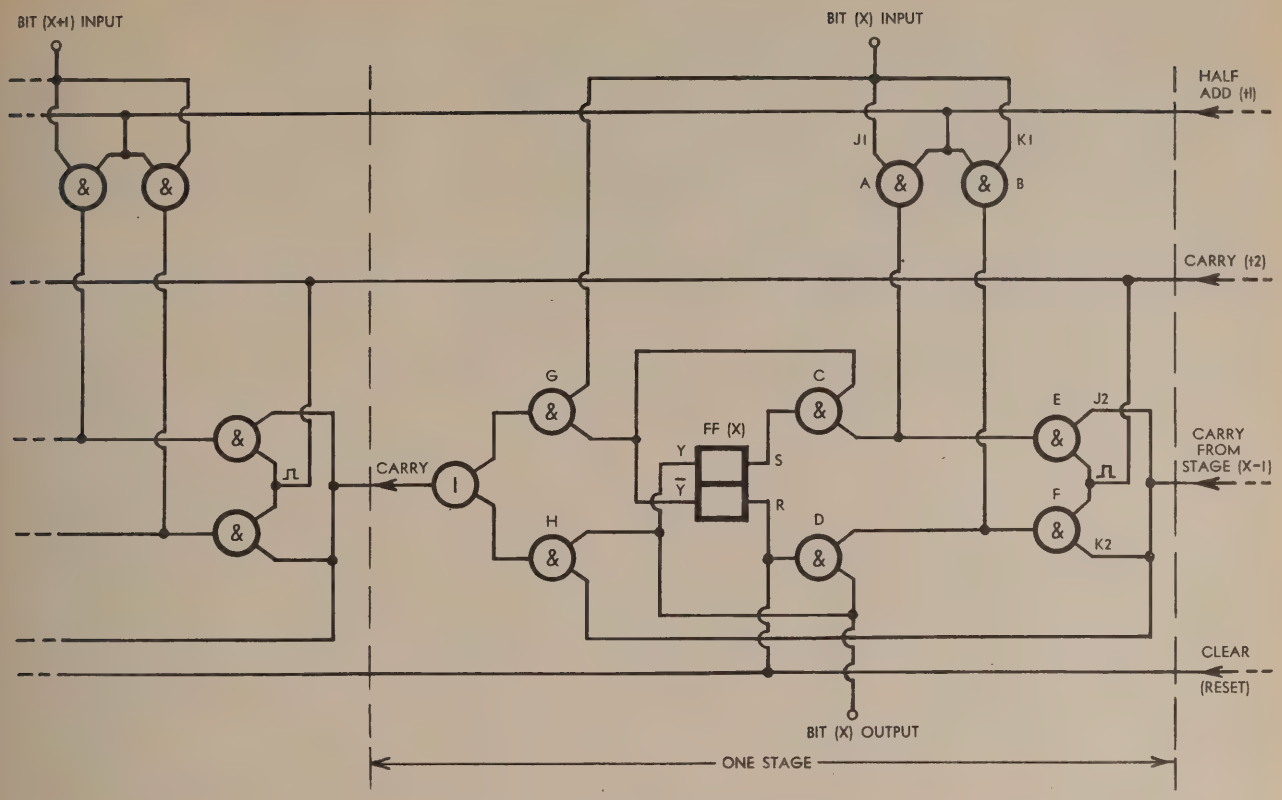


FIG 4 SECTION OF A TWO-STEP PARALLEL ADDER

the carries. These operations may be explained by reference to figure 4, which shows a section of an adder employing the two-step technique.

As may be seen, each stage consists of an R-S flip-flop together with eight AND gates and an OR gate. The six AND gates connected to the inputs of the flip-flop effectively convert it into a "double J-K" type: gates A and B provide one pair of J-K inputs, gates E and F provide a second pair, and gates C and D perform the internal self-gating necessary to produce complementation in the event of J=K=1 at either pair of inputs. The two AND gates and OR gate connected to the flip-flop outputs are used to generate the stage carry.

Operation is as follows: The flip-flops are first cleared by means of a pulse applied to the reset line. The addend is then entered into the adder flip-flops by applying a pulse to the "half-add" line while the addend bits are presented to the input terminals. An addend bit having a value of 1 causes the appropriate flip-flop to be set via gates A and C, while a bit having a value of 0 causes the appropriate flip-flop to remain in the reset state.

The augend is next applied to the input terminals, and a second pulse applied to the "half-add" line. This has the effect that, in all stages for which the augend bit has the value 1, the flip-flop is complemented **regardless of its existing state.** The reason for this is that the J1 and K1 inputs of gates A and B are connected in parallel, so that at all stages where the input is at 1 the half-add pulses are fed simultaneously to the inputs of both gates C and D.

At this juncture the state of the flip-flop of each stage of the adder represents the Exclusive-OR function of the appropriate bits of the addend and augend. It will be set to Y=1 if either of the bits alone was at 1, or reset

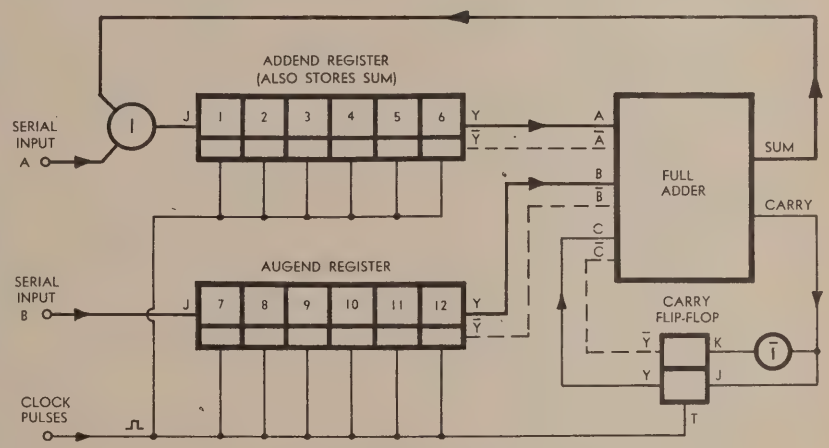


FIG 3 A SIX-BIT SERIAL ADDER

to Y=0 if either both bits were 0 or both were 1. Accordingly, the gates connected to the outputs of each flip-flop will produce a stage carry signal of 1 if **either** (the flip-flop is still set AND there is a carry from the stage next lower in significance) OR (the flip-flop is reset AND the augend bit is at 1). Reference to the truth table for a full adder given earlier should show that these are the situations for which a stage carry is required.

The addition process is completed by application of a pulse to the "carry" line while the augend bits remain connected to the input terminals. At the carry input of each stage, the gating inputs J2 and K2 are again connected in parallel; thus, wherever the carry signal from the stage next lower in significance is at 1, the flip-flop will be complemented as before, regardless of its existing state. The adder flip-flops will then contain the appropriate bits of the sum of the addend and augend in accordance with the full adder truth table. The sum may be transferred from the adder by means of the output terminals.

A parallel adder of this type is not restricted to the addition of only two numbers: further numbers may be added, simply by presenting them to the input terminals and applying pulses in turn to the "half-add" and "carry" lines. The flip-flops of the adder will continue to store the sum as it is accumulated and, for this reason, such an adder may be termed an **accumulator register.**

Although we have examined only addition and subtraction in the foregoing brief discussion of binary arithmetic circuitry, it should be realised that digital elements may be arranged to perform many other functions—multiplication, division, squaring, finding square or cube roots, exponentiation, and so on. In general the configurations used for performing these functions are developed in a similar fashion to those

The front panel of one of the smallest general-purpose digital computers produced to date, the PDP-8/S by Digital Equipment Corporation. (Courtesy Digital Equipment Australia Pty. Ltd.)

for addition and subtraction. Unfortunately space limitations will not permit even a brief discussion of these functions here; interested readers are referred to the standard computer design texts, some of which are listed in the bibliography at the end of this chapter.

Any configuration of digital elements used to perform an arithmetic operation or operations on numerical information may be called a digital "computer." However, within the very large and rapidly growing class of machines denoted by this general term, there are two rather distinct groups.

One of the groups comprises machines which consist essentially of a group of digital elements interconnected in somewhat similar fashion to the digital instruments examined in the last chapter, so that they perform either a single arithmetic operation or a group or sequence of such operations in a more-or-less fixed manner. Termed **special-purpose** digital computers, such machines are coming into extensive use in scientific, industrial process control and military applications.

Typical examples of special-purpose computer applications are the automatic control of road traffic, automatic grading of timber using digitised stress deformation data, automatic navigation, airline reservation systems, automatic control of analog computers, and automatic direction and firing of both ground and airborne artillery using digitised radar information.

The second main group of machines are those termed **general-purpose** digital computers, and it is these with which the remainder of this chapter will be concerned. It is this group of machines rather than the special-purpose machines which are normally recognised by the layman as "computers" or "electronic brains." The distinguishing characteristic of this group of machines is that they possess an effective **repertoire** of elementary logical, arithmetical and information handling operations, all of which are available repeatedly and in any order to permit their arrangement into almost any desired functional sequence.

In order to perform a particular digital operation or set of operations using a special-purpose digital computer, such a computer must usually be designed specifically for the job; consequently it will be of little or no use in performing other operations. In contrast, a general-purpose computer may be arranged to perform virtually any number of different tasks, by merely providing it each time with an appropriate "program" of instructions. It is this versatility which makes the general-purpose digital computer probably the most useful and important machine yet developed.

The operation of a modern general-purpose digital computer involves two important principles. One is the principle of functional programming, whereby the effective interconnections between, and the signals applied to, a set of digital elements are controlled so that the elements may be effectively "instructed" to assume various different configurations and perform in each case the appropriate functions. The second principle is that of the stored program, whereby the set of instructions used to direct the operation of the machine are stored within it, in sequence, permitting the machine to be arranged to repeat and modify its operations upon making logical decisions.

When a group of digital elements are interconnected, the function performed by the resulting configuration is a consequence not only of the particular elements used, but also of the configuration itself. Altering the configuration will in general change the function which is performed; conversely, a given group of elements may be made to perform a variety of functions by making appropriate changes to their interconnections. Hence, for example, a digital instrument may be provided with a function switch permitting it to be employed as either a period timer, a frequency meter or a counter, while a special-purpose computer may similarly have controls determining whether it adds or subtracts, multiplies or divides, and so on.

The important point to realise is that "making changes to element interconnections" is nothing other than a logical operation, as we have seen in earlier chapters. Consequently the interconnections between digital elements may be changed and rearranged not only by means of control switches, but also by means of any of the many other varieties of logic elements. A group of elements may therefore be arranged to perform a wide variety of functions by making appropriate interconnections by means of additional logic elements, groups of which may be activated together by control signals to form the required configurations and admit the appropriate signals to them.

The control signals used to set up and operate the various configurations effectively become **instructions** which "command" the elements to perform each of the required functions.

Certain modern digital instruments are in fact designed so that their function can be controlled in this manner. Instruments of this type are often described as "programmable," because the electrical instructions directed to them may be arranged in a sequence in order to execute a functional program.

In a general-purpose digital computer, the various control signals used to set up and operate each of the configurations in the machine's repertoire are usually derived from the output lines of decoding circuitry which is connected to a group of flip-flops or other storage elements forming the, so-called **instruction register.** The instructions for the machine then become binary numbers; each of which, when entered into the instruction register, results in the appearance at the appropriate decoder output line of control signals used to perform the corresponding function.

Thus in a very simple case "001" might produce addition, "010" might produce subtraction, "101" might result in transfer of a data number from a register A to another register B, and so on. In practice the instruction numbers are usually considerably larger than these simple examples.

Many instructions consist of two distinct parts: an order or **operation code** segment, which specifies the type of logical, arithmetic or data-number handling operation to be performed, and a second segment which specifies the location or locations of the numbers involved in or produced by the operation —the "operands."

The operation code used by a particular computer design is quite arbitrary, and it is quite rare to find the same code number used by two different machines to specify the same operation. The number of instruction bits used for the operation code depends upon the number of functionally different operations in the machine's repertoire, for each operation must of necessity have a unique order code. Many machines use three or four bits to specify the general class of operation, with additional bits to define particular variations within each class.

Computers differ considerably in the number of operand locations given in the second segment of two-part instructions. Some small machines provide for only a single operand location, while very large or very versatile machines may provide for specification of the location of up to four operands.

Early general-purpose digital computers consisted essentially of an arithmetic unit whose operation was directed, in the manner just described, by "programs" of instruction numbers fed into a control input via a medium such as punched paper tape. The instructions were performed in the same order that they were fed into the machine, and more or less instantaneously; operation was thus rather similar to that of a player piano.

Although such machines could perform arithmetic operations at a considerably faster rate than had previously been possible, they were seriously limited in operational flexibility because they could only perform calculations requiring a fixed sequence of operations. Thus, for example, they could not be arranged to follow one of two alternative courses as the result of making a logical decision. Similarly they could not be programmed to repeat a sequence of operations until a given result had occurred, because in fact the only way in which a sequence of operations could be repeated was by

feeding in the appropriate set of instructions each time the sequence was required.

That modern computers do not suffer from this serious limitation is a consequence of the adoption of the second important principle noted earlier: the principle of the stored program. With the program of instructions for the particular task to be performed stored in known and continuously accessible locations or addresses within a "memory" unit or store, the modern computer may be arranged to modify effectively its sequence of operations as it goes along, in accordance with the results it obtains in making suitably timed logical decisions. It can repeat certain sequences until a specified result is obtained, it can select one of a number of alternative courses, and it can be arranged to perform often-used sequences by means of a single set of the appropriate instructions to which it can automatically refer when required—saving the programmer from a lot of tedious repetition.

A simple block diagram of a modern general-purpose digital computer is shown in figure 5. As may be seen, it consists of five main sections: the multi-function **arithmetic/logic unit,** the **control unit,** the **memory unit** or store and the **input and output equipment** involved in the transfer of information to and from the machine.

In brief, the overall operation is as follows: The instructions, reference constants and other information (all numbers) which together form the program appropriate for the required task are prepared and encoded on a medium such as punch cards, punched paper tape or magnetic tape. The medium is then loaded into the appropriate input equipment and the program information read into the machine and stored in a specified group of consecutive addresses in the memory unit, under the direction of the control unit. The short set of instructions used by the control unit in performing this task is called the "loader," and has either been wired permanently into the machine or stored initially in a small portion of the memory unit by means of switches on the control panel of the machine.

When the program has been stored in the memory unit the machine is ready to perform the required task, providing the necessary input data is available via the input equipment and/or from the reference information stored as part of the program itself. In simple cases the input data may be fed into the memory unit along with the program; on the other hand complex commercial data processing programs may involve input data read in at various stages of the execution of the program from a variety of sources.

The control unit of a general-purpose digital computer is designed so that, when running, it normally performs only two main functional cycles; these it performs alternately and continuously. The two cycles are usually termed the **fetch** and **execute** cycles (also called "major states"). In some modern computers the control unit can perform other functional cycles in addition to "fetch" and "execute," but in virtually every case these are the main operating cycles.

The names of the main operating cycles are almost self-explanatory. During the fetch cycle the control unit sends control signals to the memory unit to "fetch" an instruction; it then enters the "execute" cycle, in which it decodes the instruction, supplies appropriate control signals to the other units in order to execute the specified operation, and readies itself for entry into a further fetch cycle. Normally it will enter a further fetch cycle and the sequence of cycles will repeat itself until either the control unit fetches an instruction which signifies "HALT" or "PAUSE," or the sequence is interrupted externally by the operator or a power failure.

To commence execution of a program, the control unit must be given the address in the memory unit of the first instruction. Then, when the "RUN" or "START" signal is given, the control unit is able to enter a fetch cycle and obtain its first instruction from that address. After execution of the instruction it will then fetch the next, execute it, and so on. As we shall see shortly, the "next" instruction fetched each time may be either that in the next consecutive address in the memory unit, or an instruction elsewhere in the program to which the control unit has been instructed to "jump"—perhaps as a result of making a specified logical decision.

During execution of a program, input data enters the system via the input equipment and is directed by the control unit to either the arithmetic/logic unit or the memory unit. In some machines input data can be transferred directly to either as desired, while in other machines all input data is transferred initially to the arithmetic/logic unit, from which it may subsequently be transferred to the memory unit as required.

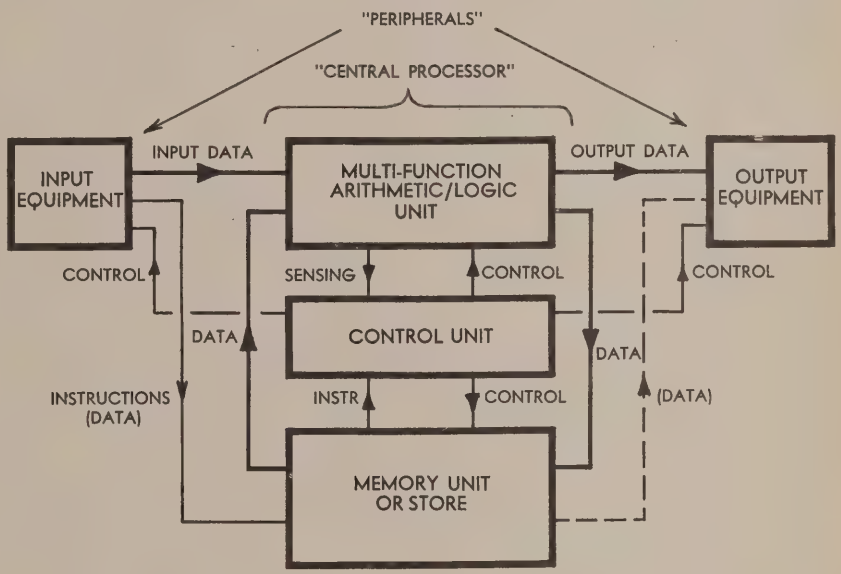


FIG 5 STORED-PROGRAM GENERAL PURPOSE DIGITAL COMPUTER

In performing the required operations on the data, the control unit transfers numbers as required from specified addresses in the memory unit to specified registers in the arithmetic/logic unit, and vice-versa. Similarly it controls the manipulation of numbers within the arithmetic/logic unit, sensing the presence, magnitude or sign of register contents as required. In some machines it may also be capable of directly manipulating numbers stored in certain portions of the memory unit. And finally, when the appropriate instructions occur, it can cause the transfer of data out of the system via the output equipment—either from the arithmetic/logic unit, the memory unit or from either depending upon the machine concerned and the requirements of the program.

With the foregoing general description as background it should now be possible to examine briefly the basic components and operation of each of the main computer sections shown in figure 5. The discussion will commence with the control unit, which might well be regarded as the "brain" of the machine.

As we have seen earlier the control unit consists partly of a storage register, the **instruction register,** into which each instruction is transferred from the memory unit during a fetch cycle. The instruction register usually consists of a set of flip-flops whose inputs are connected to the memory unit via control gates. Transfer of instructions into the instruction register may be in either serial or parallel fashion depending upon the design of the machine; most modern machines employ parallel transfer in the interests of operating speed.

Connected to the instruction register elements is the decoding circuitry required to derive control signals appropriate for each of the instructions to which the machine has been designed to respond. Part of the decoding circuitry is concerned with the operation code, being arranged to produce the master control signals used in setting up the element configurations and signal paths required in the execution of the instruction. The remainder of the decoder is usually arranged so that its output signals may be used alternatively, depending upon the general operation specified by the operation code, either to specify further the function executed, or to direct the memory unit in selecting a required operand or set of operands.

Another major section of the control unit consists of the circuitry concerned with generation and distribution of the many pulses used to activate the functions and timing of the complete machine. These functions are usually performed by a master "clock" oscillator, together with scalers, delay units and other timing elements. Associated with this circuitry is that which determines whether the machine is running or stopped, performing the "fetch" cycle or the "execute" cycle, and so on. The latter functions are normally performed by a small group of flip-flops and logic gates.

Finally, there is a most important section of the control unit—that which

keeps track of the progress of the machine as it executes the program instruction by instruction. Basically this section consists of a flip-flop register termed the **program control register** or **program counter,** which is usually arranged so that its content is used by the control unit to specify the address in the memory unit from which the "next" instruction is taken during a fetch cycle.

The program control register must contain the address of the first instruction of a program prior to the initiation of its execution. Also it is the content of this register which is modified by the control unit in response to an instruction signifying a "jump" to an instruction other than that in the next consecutive memory unit address. To enable the contents of the program control register to be set up for these purposes, the register flip-flops are provided with gated inputs connecting both to a set of switches on the control console and to logic circuitry deriving signals from the instruction decoder.

Internally the flip-flops of the program control register are interconnected as a pure-binary "up" counter whose content is normally incremented during each execute cycle by means of a single pulse derived from the pulse distribution circuitry. This ensures that in the following fetch cycle the instruction placed in the instruction register will automatically be taken from the memory unit address next consecutive to that which contains the instruction currently being executed.

The arithmetic/logic unit consists essentially of one or more flip-flop registers, together with the logic control circuitry required in the manipulation of numbers contained in, or transferred between these registers. Generally at least one of the registers is an **accumulator register** capable of accumulating the result of a serial or parallel addition of numbers. Other registers may be designed for special purposes, such as storage of the multiplier number during multiplication.

Usually at least one of the registers in the arithmetic/logic unit is a shift register capable of shifting its content in either direction. Similarly there is normally at least one register whose content bits may be complemented, either individually or as a whole, and at least one whose content may be incremented. All arithmetic registers are normally arranged so that they may be cleared (reset) when directed by an appropriate instruction.

As we have seen from the general outline of computer operation, the function of the memory unit is to store data and instructions in specified memory "addresses" from which they may subsequently be retrieved.

At this point it should be emphasised that, within a digital computer, instructions and data numbers are generally identical in form. They differ only in terms of the interpretation placed upon them by the programmer—who may, in certain cases, find it necessary or convenient to arrange that particular numbers are used at different stages of the execution of his program as both data and instructions.

Because instructions and data numbers are identical in form, it is therefore not possible to identify the content of an address in the computer memory unit as either one or the other. In most machines almost any number fed from the memory unit to the instruction register will act as an effective instruction of some sort; conversely any instruction is merely a number like any other which may potentially be manipulated in the arithmetic/logic unit. This fact permits great flexibility in the use of a general-purpose computer; it also places a responsibility on the programmer to ensure that only numbers intended as instructions are used as such.

Before proceeding with a brief discussion of computer memory units there is a small point which should be clarified with regard to terminology. It is usual in computer parlance to describe any significant grouping of bits as a **word**—a general term which includes both data numbers and instructions. In some machines all words used in the system are fixed in length (number of bits) while, in others, variable-length words may be used either for data numbers, instructions, or both.

A great many types of information storage system have been developed and are being developed for use in computer memory units. Which of these systems is used in a particular computer design —either singly, or in combination—depends largely upon the machine concerned and its intended application. Small high-speed "scientific" computers generally have memory unit requirements somewhat different from those of machines intended for large data-processing installations, for example.

There are three main storage system parameters used as criteria in the selection of a system or systems to be employed in a particular computer design. These are the storage capacity, the accessibility of the stored information and the cost.

Although in theory these parameters might be largely independent, in practice they are highly interdependent. It is generally true, for example, that systems providing the highest storage capacity per dollar provide poor accessibility of the stored information, and vice-versa. Accordingly where both high storage capacity and high accessibility are required it usually becomes necessary to employ a number of systems in combination.

The majority of general-purpose digital computers in use at the present time employ the **ferrite core memory system,** either alone or in combination with other systems. The popularity of this system stems from the fact that it offers a particularly attractive capacity/accessibility compromise at a reasonable cost. Although other systems currently under development may eventually prove more attractive than the ferrite core memory (i.e., the thin-film, plated wire, integrated semiconductor, and superconducting cryogenic memory systems) it seems likely that the former will remain popular for some time to come.

The storage capacity of ferrite core memories varies widely, depending upon the application. A memory for a small desk-type machine may have a capacity of 4,096 words (so-called "4K" capacity), each of perhaps 12 bits — 49,152 cores in all. In contrast a memory for a large machine might have a capacity of 32,-768 words ("32K" capacity), each word comprising 40 bits. The latter memory will employ no less than 1,310,720 cores.

Ferrite core memories are of the "random-access" type, as opposed to the "cyclic" type; that is, the content of all addresses is equally accessible at random, and one does not have to allow a waiting period until a desired storage address becomes available. Operating speeds of modern core memories vary from around 100 nanoseconds (read/write time) for fast low-capacity systems to around 10 microseconds for large high-capacity systems.

Basically the ferrite core memory consists of a large number of tiny ferrite toroids, each of which may be magnetised in either one peripheral direction or the other in order to store one bit of information (1 or 0). Generally the cores are arranged in matrix configurations, in order to reduce the amount of circuitry required for drive and sensing.

There are many different configurations used in ferrite core memories, each offering various advantages and disadvantages depending upon the capacity and operating speed required. One of the most popular configurations in current use is the so-called "3D" organised configuration, which is illustrated in figure 6.

The ferrite cores are grouped in a series of stacked matrix planes, where

GLOSSARY OF IMPORTANT TERMS

Accumulator Register: A register in the arithmetic/logic unit of a digital computer wherein may be accumulated the result of a series of additions or other arithmetic/logic operations.

Execute Cycle: One of the two main functional cycles of a computer, in which the machine actually performs the operation specified by the instruction word currently in its instruction register.

Fetch Cycle: The other main functional cycle, in which an instruction word is taken from a location in the memory unit and placed in the control unit instruction register ready for execution.

Instruction Register: The register within the control unit of the computer used to store an instruction word while it is decoded and the appropriate control signals directed to perform the specified operation.

Memory Address Register: The register associated with the memory unit of the computer wherein is placed the "address" word which specifies the location in memory into which an instruction or data word is to be stored, or from which it is to be retrieved.

Memory Buffer Register: The register of the computer memory unit used for temporary or buffer storage of instruction or data words either being stored in or retrieved from a memory location.

Operation Code: That segment of an instruction word of a computer which specifies the type of operation to be performed. The remainder of an instruction generally specifies the location of the operands involved.

Program Control Register: The register in the control unit of a computer whose content is used as the address of the next instruction word to be fetched and performed. For much of the time it is incremented as a counter so that the computer automatically performs a sequence of consecutive instructions.

the number of cores in a plane corresponds to the word capacity "N" of the memory, and the number of planes corresponds to the number of bits "M" per word. Each storage location or "word cell" comprises a group of M cores, in corresponding positions one on each plane. Every such group is identified by an address, so that N different address numbers are required to specify all the storage locations. This means that the number of bits required for addressing is given by the Nth power of 2.

The simple memory shown in the illustration has a capacity of 64 words each of eight bits. It thus consists of eight planes, each comprising 64 cores arranged in an 8 x 8 matrix. A six-bit address word is sufficient to permit addressing of the 64 storage locations.

The cores of a given storage location are selected electrically using a system known as coincident-current addressing, whereby each core of the memory is threaded by mutually orthogonal drive wires which effectively define it in terms of "X" and "Y" co-ordinates. Thus the cores which form a particular storage location have the same X and Y co-ordinates on each plane.

The ferrite material used in the cores has what is known as a "square loop" magnetisation characteristic, which means that in order to cause a core to adopt a stable magnetic orientation in either direction it is necessary to apply a magnetisation force in excess of a critical value. This property allows the set of cores forming a particular memory location to be selected by simply passing currents corresponding to approximately half the critical value down the appropriate X and Y co-ordinate lines. All cores other than those intended will thus experience only half the critical magnetisation force, and will remain unaffected, whereas those selected will experience the full magnetisation and will tend to switch.

Currents for the X and Y drive lines are provided by driver amplifiers which are fed from decoding circuitry connected to the outputs of a flip-flop register called the **memory address register.** Thus an address number entered into this register from the instruction register of the control unit or from the arithmetic/logic unit causes current pulses to be fed to the X and Y lines of each plane corresponding to the cores of the addressed location.

In addition to the X and Y co-ordinate lines threading the cores on all planes, each plane has two wires which thread only the cores on that plane. One of these, called the "inhibit" line, is concerned with control of the value (1 or 0) of the bit stored by the selected core of that plane during the storage or **write** cycle of the memory. The other, called the "sense" line, is used in detecting the value of the bit stored in a selected core during the retrieve or **read** cycle.

The inhibit line of each plane is connected to the output of a driver amplifier, and the inputs of the driver amplifiers are connected via gating circuitry to the outputs of flip-flops forming a register called the **memory buffer register.** It is this register which is used for temporary or "buffer" storage during both the write and read cycles of the memory.

The functioning of the memory during the write cycle is as follows: The data or instruction word to be stored is entered into the memory buffer register, while the address word specifying

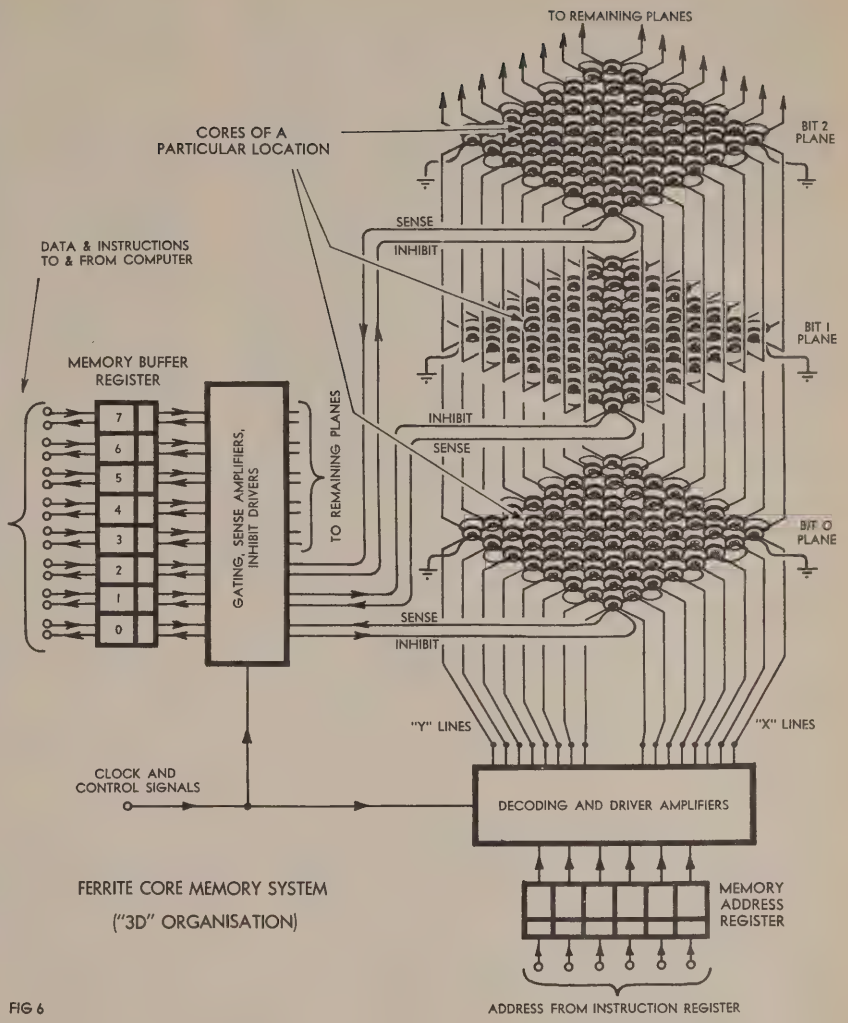


FIG 6

the required storage location is entered into the memory address register. The logic involved in the drive circuitry feeding the inhibit lines is such that, wherever a bit stored in the memory buffer register is a 0, a current pulse corresponding to half the critical magnetisation level is fed to the inhibit line of the appropriate plane at the same time as the selection pulses are fed to the X and Y co-ordinate lines. The polarity of the inhibit line pulse is such that it opposes the effect of the X and Y line pulses, thus preventing the core on that plane from switching.

By this means the cores on planes corresponding to bits in the memory buffer register of value 0 are prevented from switching, while those corresponding to bits of value 1 are permitted to do so. Hence at the conclusion of the write cycle the selected set of cores has adopted magnetic orientations corresponding to the bits of the word in the memory buffer register, and the word is "in memory."

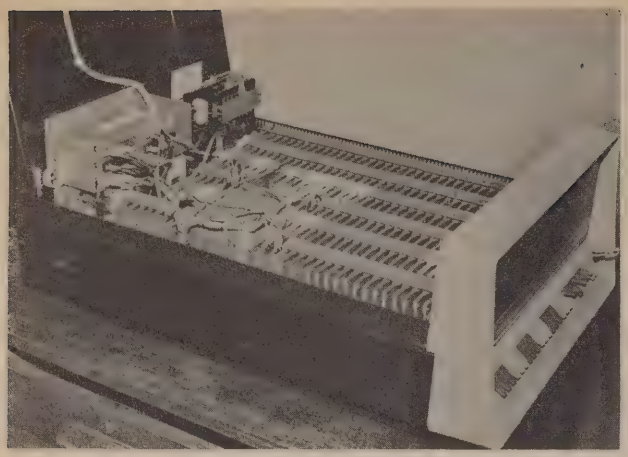
The sense line of each plane is connected to the input of a sensing amplifier and, as with the inhibit drivers, the sensing amplifiers are connected via gating circuitry to the memory buffer register. In this case it is the sensing amplifier outputs which are connected to the inputs of the memory buffer register flip-flops.

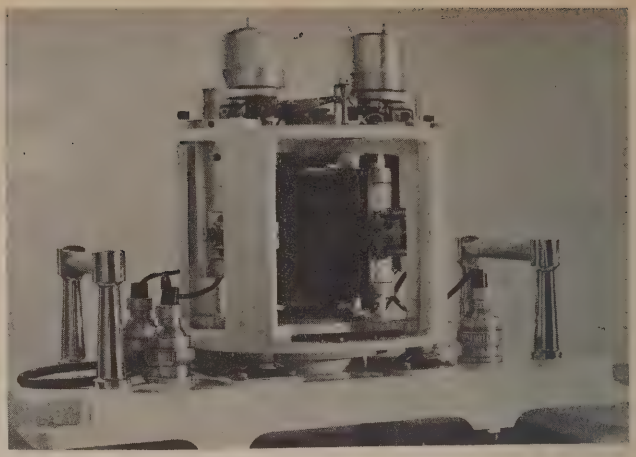
To read the word stored in a selected location, the X and Y co-ordinate lines of the location specified by the address word in the memory address register are fed with half-critical pulses as before — only in this case they are in the opposite direction. The inhibit lines are not driven, so that all cores of the selected location will receive a magnetisation force tending to switch them to the "0" direction of magnetic orientation.

Fairly obviously the only cores which will in fact switch will be those which were already storing a 1; those which were already storing a 0 will not change. Hence the sensing wires of the planes whose core stored a 1 will produce a significant induced voltage due to the sudden flux reversal, while those of the planes whose core stored a 0 will not produce such a voltage. (There will in general be some induced voltage due to transient effects in the "half-selected" cores, but this can be kept to a relatively low level.) The voltages induced in the sense lines are amplified and used to set the appropriate flipflops of the memory buffer register, and thus the word which had been stored is caused to re-appear in that register.

It may be noted that the read cycle as just described is destructive: i.e., the information which was stored in the selected location is effectively erased during the reading-out. In order to effectively "retain" the information it is therefore necessary to re-write it again. This is usually done, as it is generally required that words stored in a computer memory unit should be retained until subsequently over-written with other material.

Hence in a memory of this type the





Left: an inside view of the compact Digital Equipment PDP-8/S computer. Right: The magnetic memory drum used in the English Electric "DEUCE" computer, which appeared in 1955. (First picture courtesy Digital Equipment Australia Pty. Ltd.)

storage of information involves only the "write" cycle, while its retrieval involves both a "read" cycle and an immediately following "write" cycle.

Other magnetic core memory systems have been developed which are inherently capable of non-destructive readout. Many such systems employ multi-aperture cores or "transfluxors," which permit sampling of the stored information without permanent change to that information.

Magnetic core memories and their counterparts under current development constitute what may be regarded as rapid-access storage of moderate capacity. In general, some memory facilities of this type are required by all general-purpose computers for program storage and for temporary storage of data during computation.

In small machines, this type of storage alone may be quite adequate; many of the smaller desk-type "scientific" computers are, in fact, designed around such a memory. However machines intended for processing of bulk data or for multiple programming applications often require considerably greater storage capacity than can economically be provided by a magnetic core memory. In these cases the rapid-access facilities of a magnetic core memory are augmented by a high-capacity system or systems, usually of the cyclic-access type.

Commonly used storage systems of this type include magnetic drums, magnetic discs and magnetic tape systems, all of which are developments of the magnetic recording/replay principle employed in domestic tape recorders. Unfortunately, space limitations here prevent more than a cursory reference to such systems, and interested readers are referred to some of the standard computer texts listed in the bibliography.

The remaining main computer sections to claim our attention are those associated with information transfer to and from the computer system — the input and output sections. As the equipment which is employed in these sections is in a sense auxiliary to the control, arithmetic/logic and memory units and their functioning, it is often termed the "peripheral" equipment.

Many different types of input and output equipment are used with general-purpose digital computers; which equipment is used with a particular machine depends largely upon the application. In some commercial applications input and output are purely by means of punched paper cards; in others input is by means of magnetic tape while output is in the form of invoices and other documents produced by an automatic high-speed printer. In contrast, a small scientific machine may be provided only with a teletypewriter and slow-speed punched paper tape reader/punch.

In some scientific and industrial applications the computer itself is actually the "peripheral," forming simply a small part of a large digital monitoring and control system.

Punched cards are used both for input and output of information. They consist of rectangular pieces of stout paper or card with the required information in the form of an array of rectangular or round holes. Depending upon the particular system used, a card may be used for either a single instruction or data work, or for sequences of words. The cards are usually read into and punched under the direction of the computer using different machines, termed naturally enough a "reader" and a "punch," although combination reader/punches have recently been developed.

Cards are usually prepared for a computer system using either manually operated or automatic punching machines which are not connected to the computer system. Similarly the cards produced by a punch under computer control are subsequently processed in other independent equipment. Auxiliary equipment of this type is often called **off-line** equipment, to distinguish it from that connected directly or **on-line** with the computer.

Punched card readers typically operate at speeds of from 100 to 1,000 cards per minute, generally employing photoelectric sensing, while card punches typically operate at speeds of 100 cards per minute.

Punched cards are relatively cheap, and permit convenient alteration to programs and data: cards with errors or information to be changed can easily be withdrawn from a stack and replaced. However, preparation of information in card form can be tedious and time-consuming, as can be the subsequent retrieval of the result of computation from the output cards.

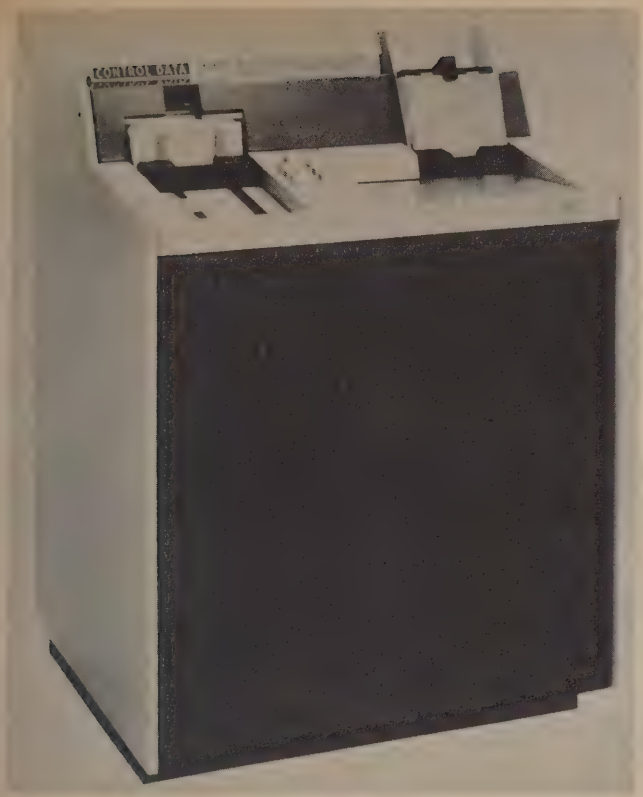
Like punched cards, **punched paper tape** is used both for input and output of information. It consists of a paper ribbon approximately 1 inch in width, containing the information as round holes punched in a number of longitudinal rows — from five to eight rows being used, depending upon the particular system employed. Information transfer to and from the computer is via a reader and a punch as before; transfer rates vary depending upon the type of equipment used. Readers employing mechanical sensing operate at typical speeds of 10-30 characters per second, while those employing photoelectric sensing are capable of operation up to 1,000 characters per second. Punches typically operate at speeds of 110 characters per second.

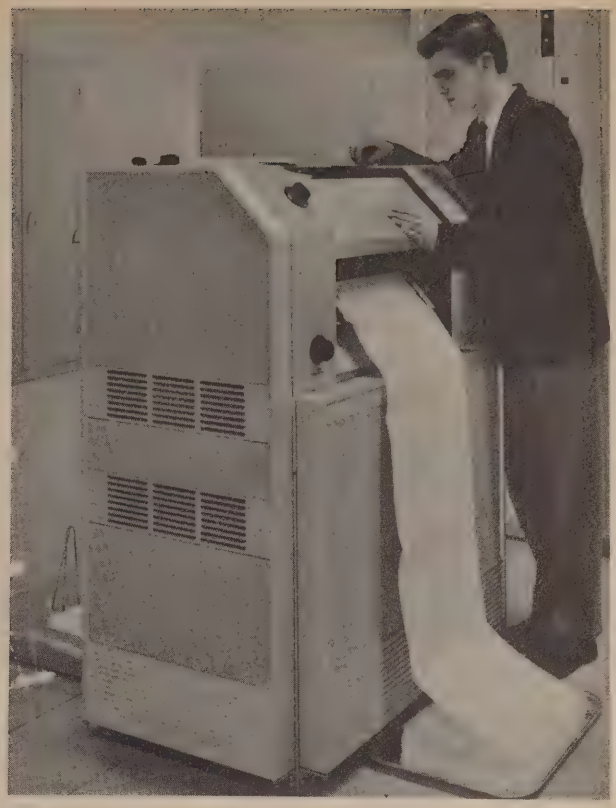
Magnetic Tape can be used for both input and output, being on extremely fast and efficient medium. The same equipment used for magnetic tape memory may in general be used for input and output — in fact the distinction between "memory" and "input-output" becomes somewhat blurred in magnetic tape systems, because a reel of tape used to receive the "output" of one calculation may often become the source of "input" for another, becoming in effect part of the system "memory." Typical magnetic tape units operate at transfer rates of 20,000 characters per second.

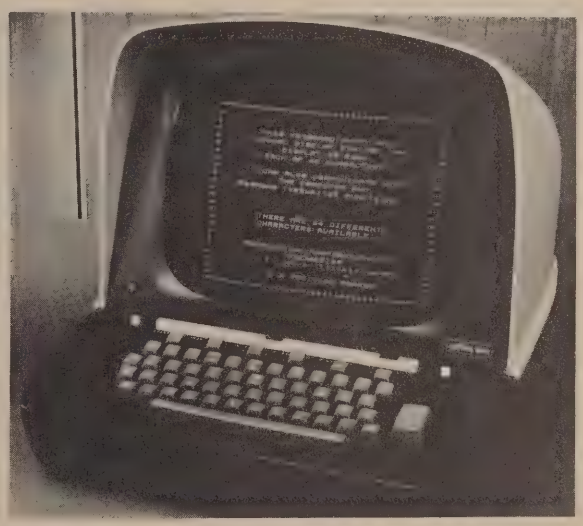
High-speed printer units are machines used to produce printed invoices and other documents either directly from computer output on-line, or off-line from tape machines or card readers. A number of different operating systems are used, some employing type drums or chains operating at high speed in conjunction with solenoid-operated hammers, while others employ electrostatic or photographic printing techniques. A typical machine is capable of producing up to 1,500 printed lines per minute, each with up to 132 characters.

Teletypewriters are often used for direct operator-machine communication, particularly on smaller machines. The keyboard may be used for input to the machine while the typing mechanism may be used to deliver output from the machine. This type of input-output equipment is very inefficient, operating typically at speeds of 10 characters per second or less, but in some applications this disadvantage is outweighed by the flexibility of the direct man-machine relationship. Teletypewriters used with small machines are often fitted with a low-speed paper tape reader and punch.

The recently developed **cathode-ray-tube or CRT display unit** has the advantages of direct man-machine communication also, but in this case combined with high operating speed and







Examples of computer input-output equipment. At upper left is the Control Data 9280 punched-card reader/punch. Above right is a General Electric high speed line printer. Below left is a Teletype 33-ASR unit complete with low-speed paper tape reader and punch, while at lower right is a Control Data CRT display unit.

efficiency. Despite its high cost it is becoming quite popular in both scientific and commercial applications, as it lends itself to the handling of both alphanumeric and graphical/pictorial information. In its basic form the CRT display unit is an output unit, but it may be also used for input by means of a hand-held photoelectric detector or "light pen."

In its simplest form the CRT display unit consists of a cathode-ray tube with precision scanning circuitry driven from a pair of digital-to-analog convertors (DACs), together with logic and control circuitry. Signals may be applied to the DACs from the computer either directly or via appropriate buffer registers to set up the specified deflection currents, following which a "bright-up" pulse is applied to the tube grid to produce a spot of light at the desired screen coordinate.

More elaborate display units employ auxiliary electronics to generate lines, circles and other geometric shapes automatically when required. Some employ small magnetic core memories with prewired codes, or alternatively special character-generating tubes, to produce auxiliary deflection signals capable of automatically producing any alphabetic or numeric character at the specified point on the display screen.

It is not uncommon for CRT display units to become so elaborate that they are provided with their own small general-purpose computer, which directs their detailed operation in accordance with general instructions from the main computer system!

Computers intended for use in scientific applications are often provided with very flexible input-output interconnection or "interface" facilities, whereby they may be connected to a variety of digital and other equipment. Thus a computer of this type may be arranged to accept as input the signals from digital voltmeters, counters and frequency meters, while its output may be arranged to adjust reaction temperatures, control the speed of motors, adjust flow rates, and so on.

To conclude this somewhat abbreviated introduction to general-purpose digital computers the following brief notes regarding programming languages may be found worthwhile.

As we have seen earlier in this chapter, the functions performed by a general-purpose computer are directed by means of binary numbers which are

interpreted by the machine as instructions. A set of these instructions is stored in the computer memory unit as a "program", which is executed by the machine when the address of the first instruction is deposited in its program control register and the "RUN" signal given.

A computer may thus be made to perform almost any given task by simply working out the sequence of elementary logical, arithmetic and transfer operations necessary for its execution and determining the appropriate machine instruction numbers for these operations. A program prepared in this form is said to be written in **machine language.**

With the earliest computers, machine language programming was the only technique available. As one might expect, it proved extremely tedious and time-consuming, and this led computer designers and users to seek programming techniques which were somewhat more rapid and convenient. It was soon realised that the computer itself could be used to perform translation of programs from "man-orientated" languages into its own machine language, simply by encoding the original program language into numbers which may be manipulated by the machine.

Developed from this realisation have been two general types of "man-orientated" programming languages, and it is in these languages that most modern computer programs are written. The two types of programming language are called respectively "symbolic" or "assembly" languages and "task-," "problem-" or "procedure-orientated" languages.

A program **assembly language** consists of a vocabulary of symbols which may be used in prescribed ways as both mnemonics for the machine language instruction numbers and as translation instructions for a program called the **assembler program** which is used to direct the computer in translating the mnemonics into their equivalent machine language instructions. The symbols usually consist of the standard alphabetic and numeric characters together with spaces and other punctuation characters.

Assembly language programming allows the programmer to write his program as easily remembered and manipulated mnemonic symbols, and to assign convenient arbitrary labels to memory storage locations often used during a calculation. Thus addition of the contents of a particular memory location labelled "DATA1" to an accumulator register symbolised as "A" might be written

AD A DATA1

where the mnemonic "AD" is used to symbolise the operation of addition.

Translation of the assembly program into the machine language "understood" by the computer is performed by the computer itself, under the control of the assembler program supplied with each machine as part of its so-called "software package." The assembler program —which is itself a machine language program—is fed into the machine, and under its direction the encoded assembly language program is translated into machine language. In general the translation is on a one-for-one basis; that is, one machine language instruction number is produced for each symbolic instruction.

As an example the sample assembly language instruction given above might be translated by the assembler program into the machine language instruction

101110110110110

where the first four bits on the left might represent the operation code, in this case signifying addition; the next three bits might represent the address code of the arithmetic/logic accumulator known to the assembler program as "A"; and the final eight bits might signify the address of the memory location to which the programmer has arbitrarily assigned the assembly language label "DATA1."

Because the symbols used by a programming assembly language must correspond to operations which may be performed by a particular computer, and because no two computers perform exactly the same basic operation in exactly the same way, in general every computer has its own assembly language. Thus a program written in the assembly language of one machine must usually be re-written if it is to be made suitable for translation by the symbolic assembler program associated with another machine. This is one of the disadvantages of programming assembly languages..

A second disadvantage is that assembly programming is still rather tedious in that a separate instruction must still be written for each individual elementary operation to be performed by the computer.

Procedure-orientated languages are the result of attempts to obviate these disadvantages. They are arranged so that a program may be written in a form which is largely independent of the structure and organisation of a particular machine, using symbols and constructions which are as close as possible to those of the mathematical, scientific, engineering or commercial context of the computations concerned.

Probably the three most commonly used procedure-orientated programming languages at present are called **FORTRAN, ALGOL** and **COBOL.** The first and second of these are both algebraic programming languages that employ a notation rather similar to everyday mathematical notations used for expressing and solving problems in mathematics, science and engineering. Fortran (formed by ellipsis from "Formula Translation") was developed in the U.S., while Algol (from "Algorithmic Language") originated in Europe. Today both enjoy virtually international use.

Cobol (from "Common Business-Orientated Language"), like Fortran, originated in the U.S. It is a language designed to provide a basic set of procedures for commercial data processing applications.

Many other procedure-orientated languages are in use, including some designed for refined electronic circuit analysis techniques. New languages are continuously being developed, some intended for general-purpose use and others for specific applications. Currently under development in the U.S. is a new general-purpose language called PL/1 (for "Programming Language, version 1"), which combines many of the features of Fortran, Algol and Cobol.

Like assembly language programs, procedure-orientated language programs must be translated into the corresponding machine language programs "understood" by particular computers. This translation is again performed by each computer itself, under the direction of programs as before supplied by the computer manufacturers as part of the "software package." The translation process involved is somewhat more complex than that for an assembly program as, in general, an instruction statement in a procedure-orientated language is equivalent to not one, but a complete sequence of elementary machine language instructions. For this reason the programs used for procedure-orientated language translation are usually called **compilers** rather than assemblers.

It has only been possible in this chapter to discuss the most basic aspects of general-purpose digital computers and their operation. There are a great many important topics which, from sheer necessity, have had to be omitted entirely; for example no mention is made of "program-interrupt" techniques for efficient input-output transfer, of the use of "executive", or "monitor" programs to supervise the operation of a complex computer system, or of multiple programming techniques such as time-sharing. Other topics have been mentioned only briefly. It is hoped that despite these limitations the reader will have gained sufficient background to support further study, possibly by means of some of the books listed in the attached bibliography.

SELECTED BIBLIOGRAPHY

DAKIN, C. J., and COOKE, C. E., **Circuits for Digital Equipment,** 1967. Iliffe Books Ltd., London.

DIMITRY, D., and MOTT, T., **Fortran IV Programming,** 1966. Holt, Rinehart and Winston, Inc. New York.

EVANS, D. C., "Computer Logic and Memory," in **Scientific American,** V.215, No. 3, September 1966. (Many other articles in same issue.)

FREDKIN, E., "The Time-Sharing of Computers," in **Computers and Automation,** V.12, No. 11, November 1963.

JACOBWITZ, H., **Electronic Computers Made Simple,** 1967. W. H. Allen and Co., London.

LEDLEY, R. S., **Digital Computer and Control Engineering,** 1960. McGraw-Hill Book Co., New York.

LIVESLEY, R. K., **Digital Computers,** 1957. Cambridge University Press.

MARTIN, E. W., **Electronic Data Processing,** 1965. R. D. Irwin Inc., Homewood, Illinois.

RAJCHMAN, J. A., "Integrated Computer Memories," in **Scientific American,** V.217, No. 1, July 1967.

ROWE, J., "Magnetic Core Memory Matrices," in **Electronics Australia,** V.27, No. 6, September 1965.

SAY, M. G., (Ed.) **Analogue and Digital Computers,** 1960. George Newnes Ltd., London.

SCOTT, N. R., **Analog and Digital Computer Technology,** 1960. McGraw-Hill Book Co., New York.

Chapter 12

Completing the Demonstrator

Description of a demonstrator unit which may be used both as a tuitional device and as a design breadboard for low and medium speed digital circuitry.

In one or two of the earlier chapters we gave a partial description of a demonstrator unit which the author has developed as a digital teaching device and breadboard unit to assist in the design of low-speed logic circuitry. As space problems prevented the completion of the description in later theory chapters, this will now be done.

In order that the description of the unit given here should be reasonably unified, the information given in the early chapters will be repeated; the present chapter will therefore contain all the published constructional information. While this could be criticised from the viewpoint of wastage of valuable space, the repetition is believed to be worthwhile in terms of convenience.

In commencing the description of the unit as a whole perhaps the first thing which should be noted is that it has been designed primarily as a teaching tool. The aim has been to produce a system capable of demonstrating the principles of circuit logic and digital techniques, at a cost which it is hoped will lie within the range of all interested in such teaching facilities.

To this end the circuitry employed has been kept as simple as possible and, in all cases, we have tried to use the cheapest and most readily obtained components. No attempt has been made to use exotic circuitry or devices in an attempt to push performance to the present "state of the art." Hence relative to modern digital circuit techniques the unit must be regarded as somewhat primitive.

While the unit cannot therefore be regarded as the basis of practical high-performance digital circuitry, it may nevertheless prove of value as a breadboarding system to assist in the design of low and medium-speed circuitry. Most of the circuits used have been found to operate well at speeds in excess of 20KHz, and the flexibility which has been incorporated in the interests of effective demonstration should make the unit equally suitable for the mocking-up of a variety of configurations.

As may be seen from the photographs the prototype demonstrator has been constructed using metalwork from the well-known "Lektrokit" range. We selected prefabricated metalwork for the project on the basis that it gives a polished and businesslike appearance while involving a minimum of skill in fabrication.

The demonstrator circuitry is constructed on four full-width single height Lektrokit panels, code LK-401. The four panels are mounted in a small Lektrokit rack assembled from the Kit No. 1 (code LKR-5011). To facilitiate mounting in the rack each panel is fitted with side plates (code LK-301, 2 required per panel) and rack adapter brackets (code LK-601, 2 required per panel). In addition the two lower panels are fitted with chassis rails to support full-width wiring strips (code LK-201, 2 required per panel), and the lowest panel uses two No. 1 chassis plates (code LK-111) to support the power supply components.

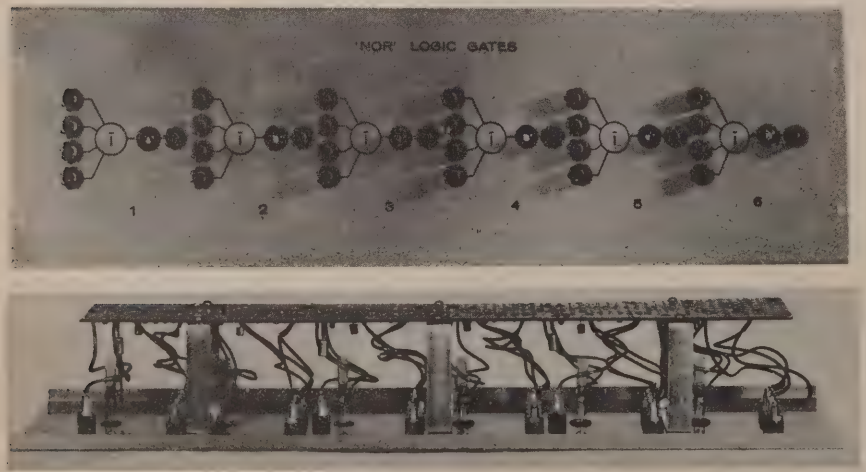
Most of the wiring of the prototype is achieved using "Vero-board" copper conductor laminate strip. Three Vero-board strips coded 4/1001 are required, one being cut approximately in half lengthwise to produce half-width strips for the two upper panels. A full strip is used for the third panel, while the fourth panel uses an 8in length which is cut from the third strip.

Both Vero-board and Lektrokit components are imported into this country by E.M.I. (Australia) Pty. Ltd., and all parts from these lines used in the demonstrator may be ordered from these firm or their agents via the usual parts suppliers. In N.S.W. orders may be placed with Watkin Wynne Pty. Ltd., of P.O. Box 318, Crow's Nest, who are the State distributors.

The uppermost of the four panels provides six logic gates of the type shown in figure 1. As may be seen, the gates use simple resistor-transistor logic (RTL), and include a small lamp which indicates the positive output condition.

The gates are nominally labelled "NOR" elements as they perform the NOR operation in positive logic (1=+9V, 0=0V). However, as we have seen in earlier articles of this series the logical function of a circuit or device may be interpreter in a number of ways, according to the assigned logic polarity convention.

Hence if the negative logic convention is used (1=0V, 0=+9V), the gates will perform the NAND operation; again, with the positive-negative and negative-positive mixed polarity conventions they will perform the OR and AND operations respectively. And if a single input is used with either the positive or negative conventions, the gates perform the NOT operation. In other words, any of the basic logical operations may be performed by assigning the appropriate polarity convention.



Above are front and rear views of the top panel of the demonstrator, which provides six "NOR" gates. The circuit used is at above right.

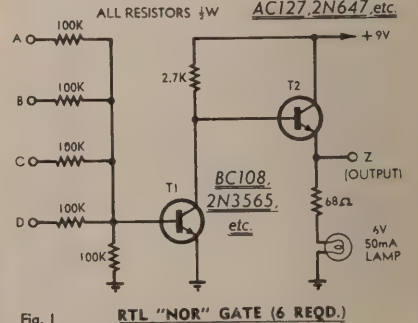
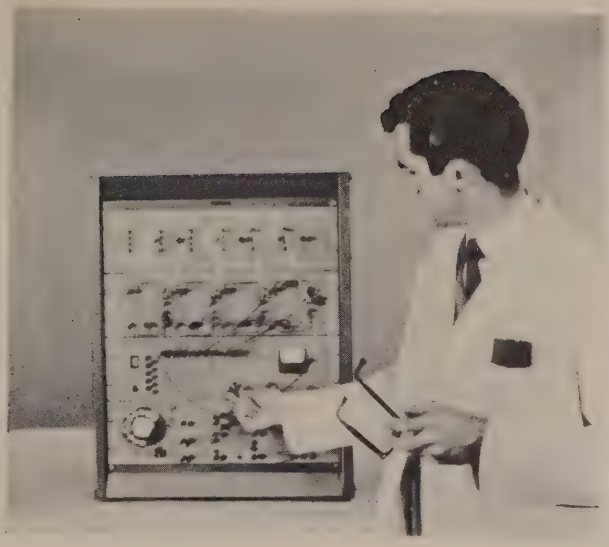


Fig. 1 RTL "NOR" GATE (6 REQD.)

CONSTRUCTORS, PLEASE NOTE

Glossy prints of the demonstrator panels are available via the Information Service at 50c each



The unit set up for demonstration of analog - to - digital conversion.

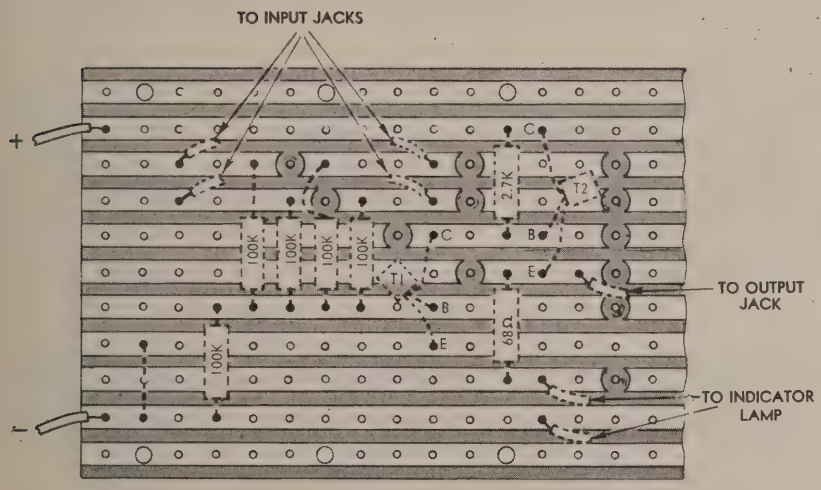
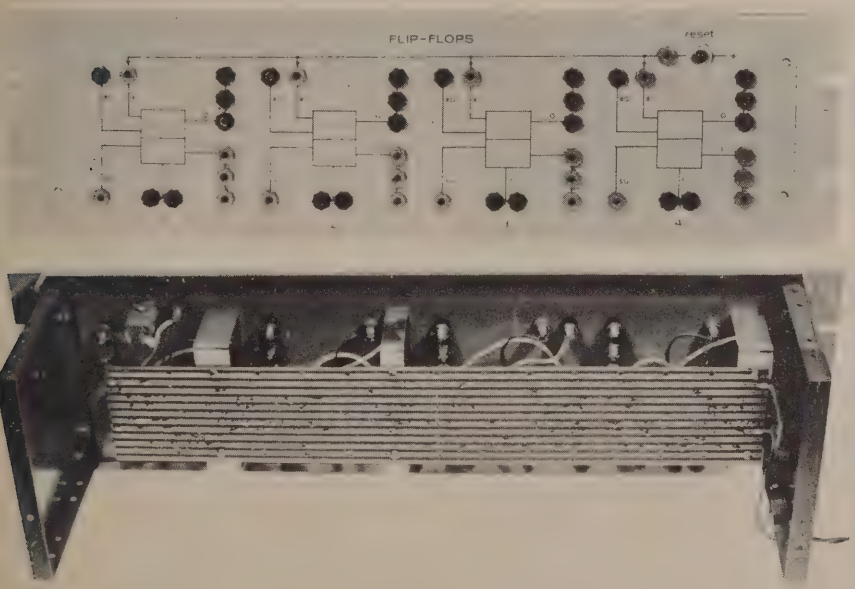


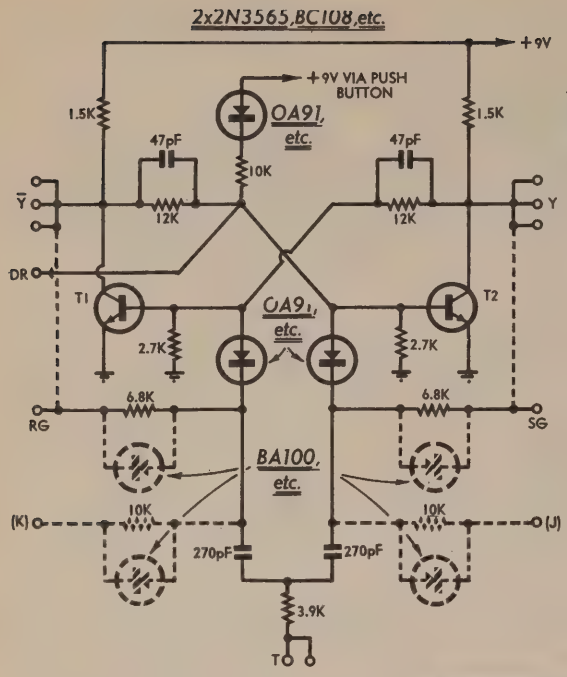
Fig. 2

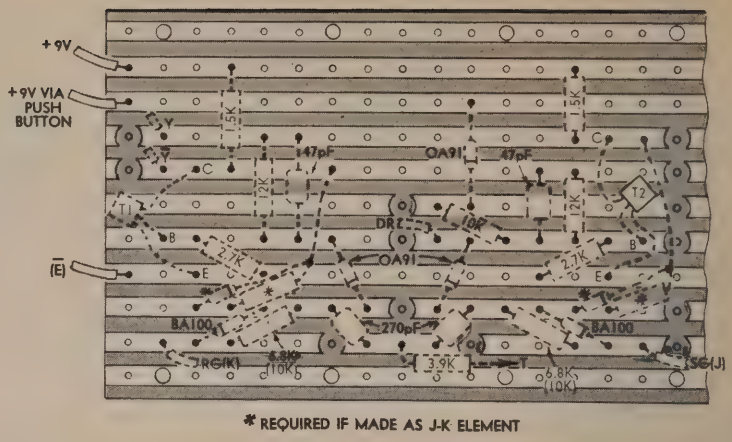
WIRING FOR ONE "NOR" GATE

Above is the wiring pattern used for a single gate on the "NOR" panel. Six of these gates are used, each wired as above.



Front and rear views of the second panel, which provides four flip-flops of either the gated R-S or J-K variety as desired.





* REQUIRED IF MADE AS J-K ELEMENT

PANEL FLIP-FLOP (4 REQD.)

At left is the circuit used for the panel flip-flops, while above is the wiring pattern. The four flip-flops used will normally have identical wiring patterns.

It should perhaps be noted that out put lamp will only light for "true" (1) output when the output polarity convention is positive. Hence for the negative and positive-negative mixed conventions the lighting of the lamp must be interpreted as signifying that the output is false (0).

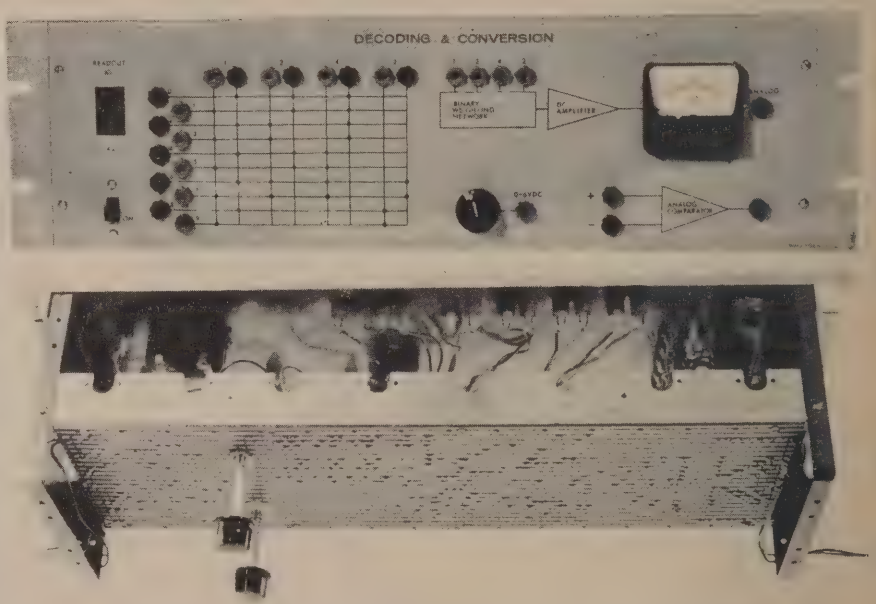
A further use for the gates from the viewpoint of the demonstrator as a whole is that they may be used purely as binary readout devices. Thus for example they may be used to demonstrate the state of the flip-flops on the second panel, or the output value of any of the other elements.

The only point to note in this regard is that, as indicators, the gates involve logical inversion; so that the lamp lights when the point to which the gate is connected is at 0V. Hence, to arrange the gates to indicate the "set" (Y=1) state of the flip-flops, where the latter are assigned the positive convention, they would be connected not to the flip-flop Y terminals but to the Y-complement terminals.

Each gate uses seven half-watt resistors, two NPN transistors and a 6V 50mA lamp of the "No. 2 Post Office Switchboard" variety. One of the transistors (T1) is a low-cost general purpose silicon planar type BC108, 2N3565 or similar, and the other (T2) is a low-cost medium power germanium type AC127, 2N647 or similar.

The input and output connections for each gate are made available on the front panel via small "banana" jacks, as may be seen. The lamps are mounted to the panel adjacent to their corresponding output jacks using close-fitting 5/16 in rubber grommets. In the prototype, red jacks have been used for the input terminals of each gate, and black jacks for the output terminals.

The wiring of the gates should be fairly clear from the diagram of figure 2, which shows a single element. Note that the strip of Vero-board used for the gates is 11 conductors wide, and that the second conductors from either side are used as supply rails for all six elements. (As mentioned earlier the strip used for the gates is cut from one of the 4/1001 stock strips, the remainder of which is used for the second panel wiring.)



Front and rear views of the third panel, which provides circuitry for BCD-decimal decoding, decimal readout, and digital-to-analog and analog-to-digital conversion.

The wiring strip is fastened to the front panel by means of three small "U" brackets which are clamped to the panel by three of the "output" banana jacks. The brackets are bent up from scraps of sheet aluminium approximately 8in x ¼in. The construction should be apparent from the photographs.

The front panel is provided with logic symbols and appropriate lettering using indian ink and wax-transfer "rub-on" lettering. To protect the markings from wear, the panel is sprayed with clear lacquer before the jacks and lamps are fitted.

The second panel of the demonstrator provides four flip-flops which use the circuit shown in figure 3. As may be seen the flip-flops may be wired as either gated R-S or J-K types, the latter version requiring additional resistors.

The number of flip-flops has been fixed at four as this is the smallest number which permits demonstration of all common counting and shifting configurations. A larger number could, of course, be used and this may be contemplated if cost is of secondary consideration; however, for most purposes, four will be found quite satisfactory. A smaller number would limit the flexibility of the demonstrator quite drastically and should not be considered except in special circumstances.

The circuitry used in the flip-flops is again based upon simple resistor-transistor logic. Commutating capacitors (47pF) are used to extend operation to about 50KHz, which limit may be extended further if recharging diodes are used across the gating resistors. The transistors used are again NPN silicon planar general - purpose types BC108, 2N3565 or similar.

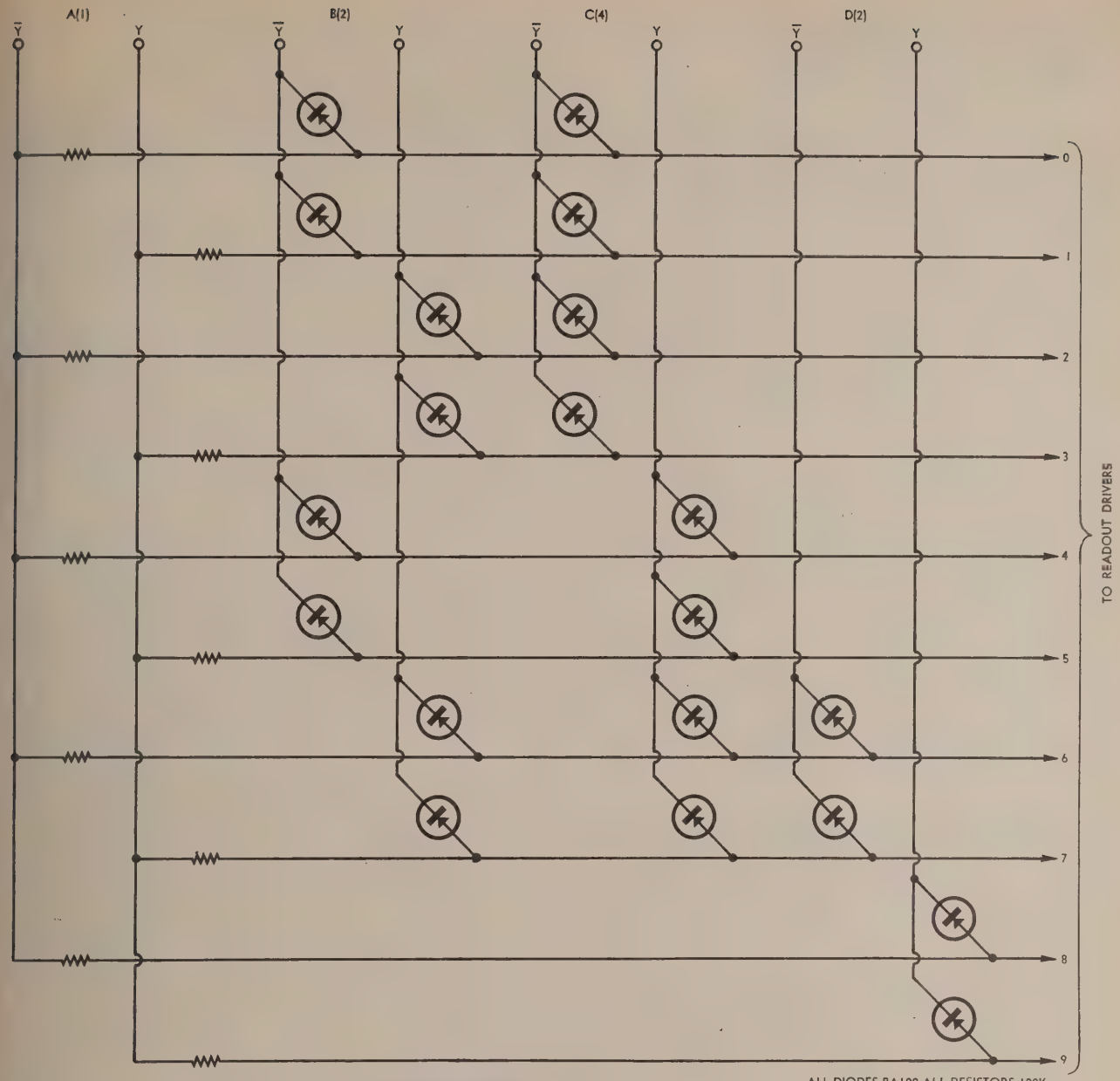


Fig. 5 **DECODING MATRIX**

When the flip-flops are wired as gated R-S types, the gating terminals are marked "SG" and "RG," and 6.8K gating resistors are used—with or without recharging diodes as desired. Alternatively, if the flip-flops are wired as J-K types, the 6.8 K resistors are taken internally to the output terminals and additional 10K resistors taken to the gating terminals which are now marked "J" and "K." If recharging diodes are desired for extended frequency range, four will be required for each J-K flip-flop.

As with the first panel, the flip-flops are wired on a Vero-board strip with their connections made available on the front panel via small banana jacks. In this case multiple jacks are used for the "T" and output terminals to facilitate interconnections between elements. Red jacks are used for the Y, SG (J) and DR connections, and black jacks for the remainder.

The four flip-flops are connected to a common reset line via 10K resistors and isolating diodes, and may be reset as a group using either a push-button or a pulse applied to a common reset terminal wired to the line. In addition each flip-flop is fitted with a direct reset (DR) terminal of its own to permit individual resetting.

No series resistance is used for the DR inputs in order that they may be used for pulse resetting, using pulses derived from other flip-flops — for example, in the demonstration of feedback BCD counting.

As it will occasionally be necessary to perform individual direct resetting of flip-flops (for example, when it is required to set initial conditions for the demonstration of a shift register or nor-

Above is the circuit of the decoding matrix, and at right that of a readout driver. Ten of the latter are required, one for each decimal output.

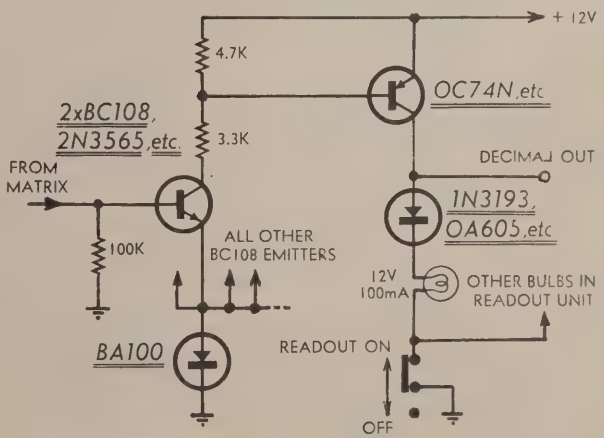


Fig. 6 **READOUT DRIVER (10 REQD.)**

At right is the wiring pattern used for the decoding and readout section of the third panel, which occupies the left half of the wiring strip.

mal ring counter) it should be noted that this MUST NOT be done by simply connecting the DR terminals to the +9V line. Because there is no protective resistance in series with the DR terminals, this will result in damage of transistor T2.

To allow direct resetting of individual flip-flops to be done safely the fourth panel of the demonstrator includes two terminals which are connected to +9V via 10K resistors. The flip-flop DR terminals may be connected to these terminals via one of two push-buttons which are also provided on the fourth panel.

Figure 4 shows the wiring arrangement used for the flip-flops. Note that the Vero-board strip used is 11 conductors wide, and that the second and third conductors from the top and the fourth from the bottom are again used as common supply lines for the four flip-flops. These are used for +9V, common reset and earth respectively.

The general construction of the flip-flop panel is very similar to that of the NOR gate panel. As before, the wiring strip is fastened to the front panel using small aluminium "U" brackets clamped under three of the banana jacks. The panel symbols and lettering are again achieved using indian ink and rub-on letters.

The third panel of the demonstrator provides facilities for the demonstration of 2421 BCD-decimal decoding, decimal readout, and digital-to-analog and analog-to-digital conversion. The left half of the panel is used for decoding and readout circuitry and the right half for conversion circuitry.

The circuit of the matrix used for 2421 BCD-decimal decoding is shown in figure 5. As may be seen, it uses both resistors and diodes in a positive-logic AND configuration. An AND decoder has been used in order that positive-logic electrical decimal outputs may be available in addition to the visual readout; resistors have been used for the "1" connections to keep costs to a minimum.

The eight BCD inputs to the matrix are connected to a horizontal row of banana jacks along the top of the panel symbol, with red jacks for the Y input connections and black jacks for those connecting to the Y-complement inputs. The ten decimal outputs of the matrix are taken in turn to the readout driver amplifiers, whose outputs are taken to both the readout unit and a double vertical row of jacks to the left of the panel symbol. Red jacks are used for the odd decimal outputs and black for the even outputs.

Ten driver amplifiers are used, one for each decimal digit, and all are identical. Figure 6 shows the circuit used. As may be seen it is a simple two-stage complementary DC amplifier using a low-cost NPN silicon planar transistor type BC108, 2N3565 or similar and a low-cost medium power PNP germanium type OC74N or similar.

The silicon diode in the first stage emitter circuit is a single component which is shared by the first stages of all ten drivers. Its purpose is to provide a small amount of cut-off bias to ensure that non-selected drivers do not conduct due to residual voltages which may be produced by the matrix. Such residual voltages tend to be produced due to the non-zero voltage drops across saturated flip-flop transistors and decoding diodes.

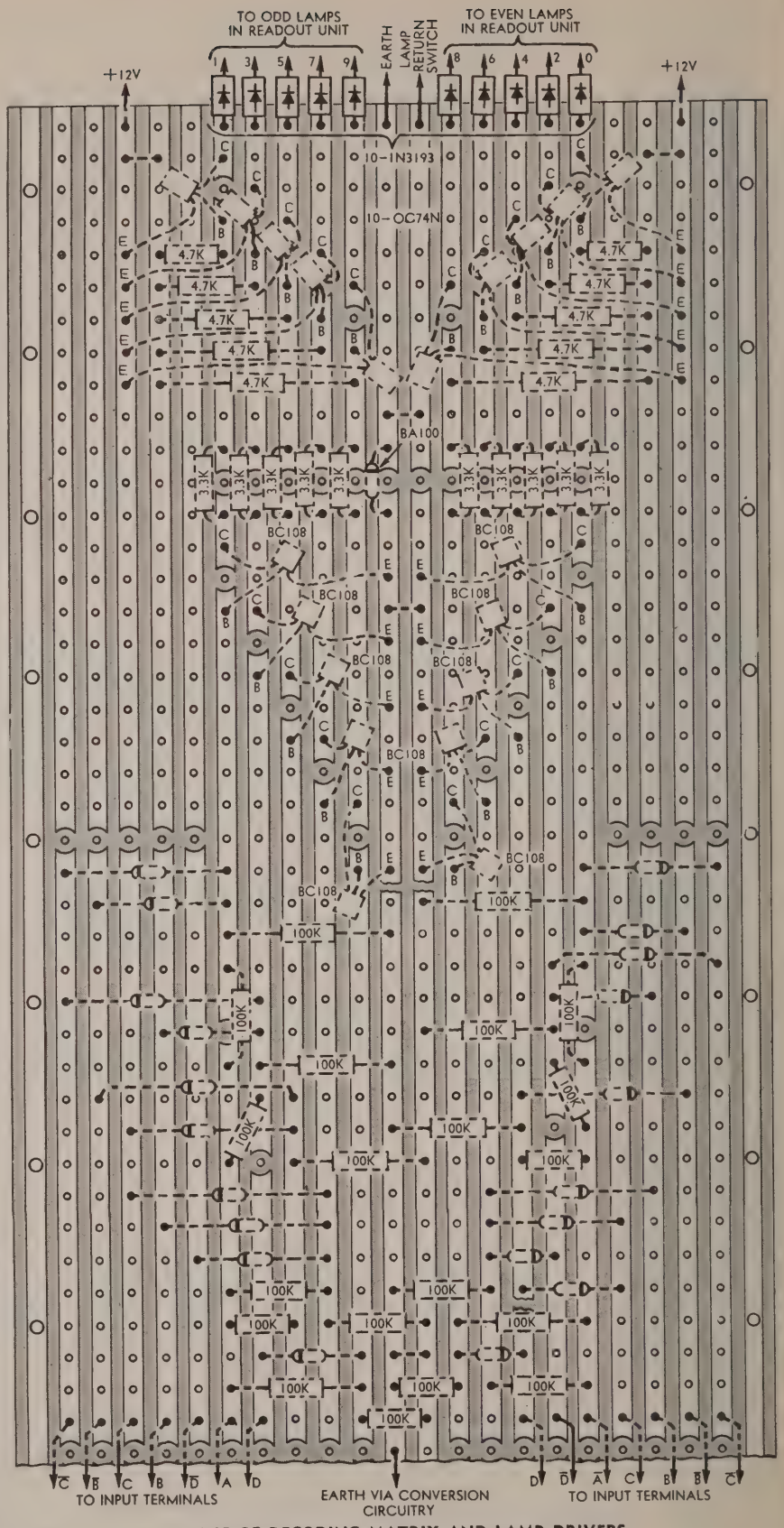


Fig. 7 **WIRING OF DECODING MATRIX AND LAMP DRIVERS**

The output of each driver connects via a power diode to the appropriate lamp of the readout unit. The common lamp return of the readout unit is taken in turn to a slide switch so that the readout may be disabled when not required. The diodes in series with each lamp prevent electrical interaction between

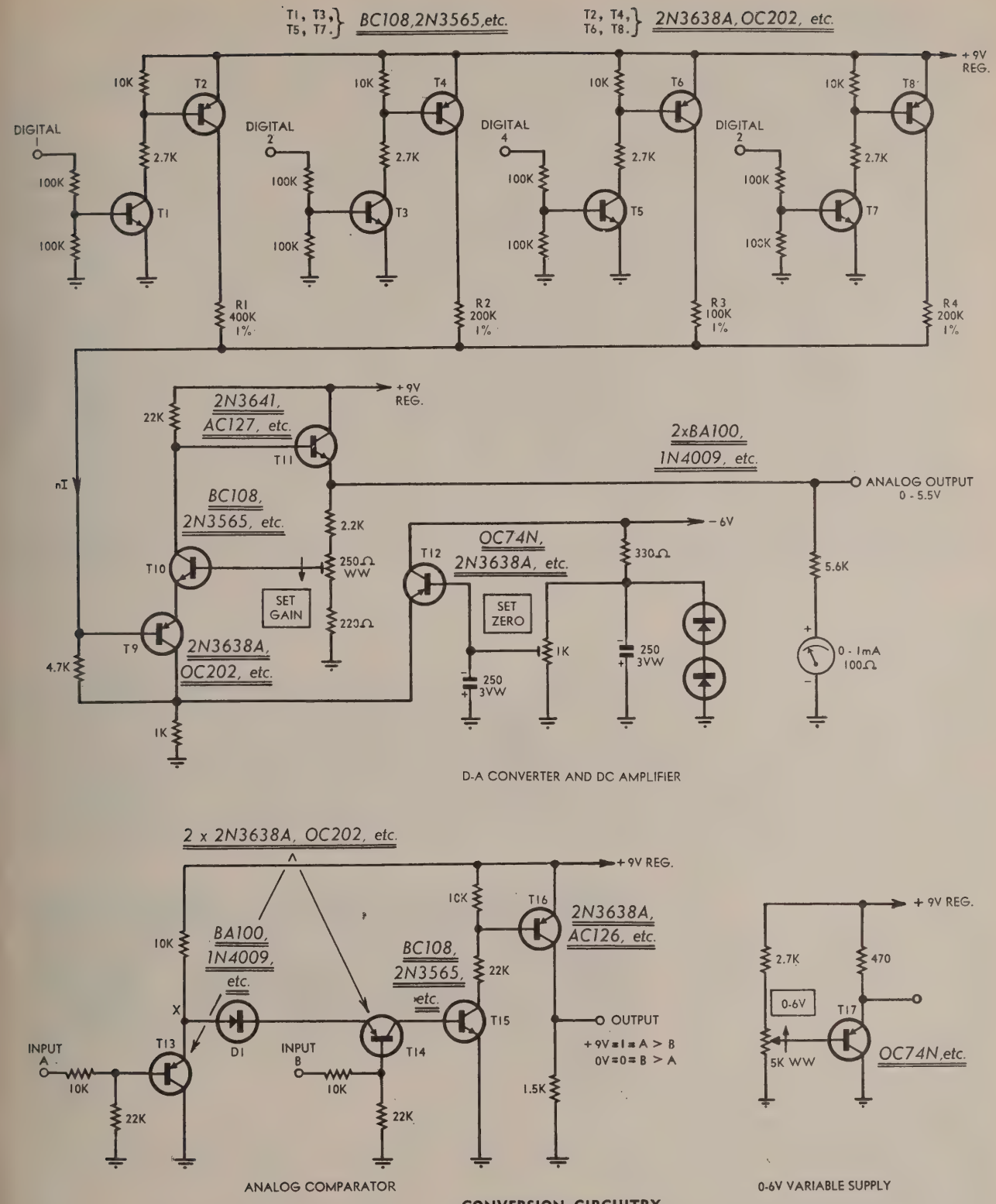


Fig. 8

CONVERSION CIRCUITRY

Above is the circuitry used for the digital-to-analog and analog-to digital conversion section of the third panel.

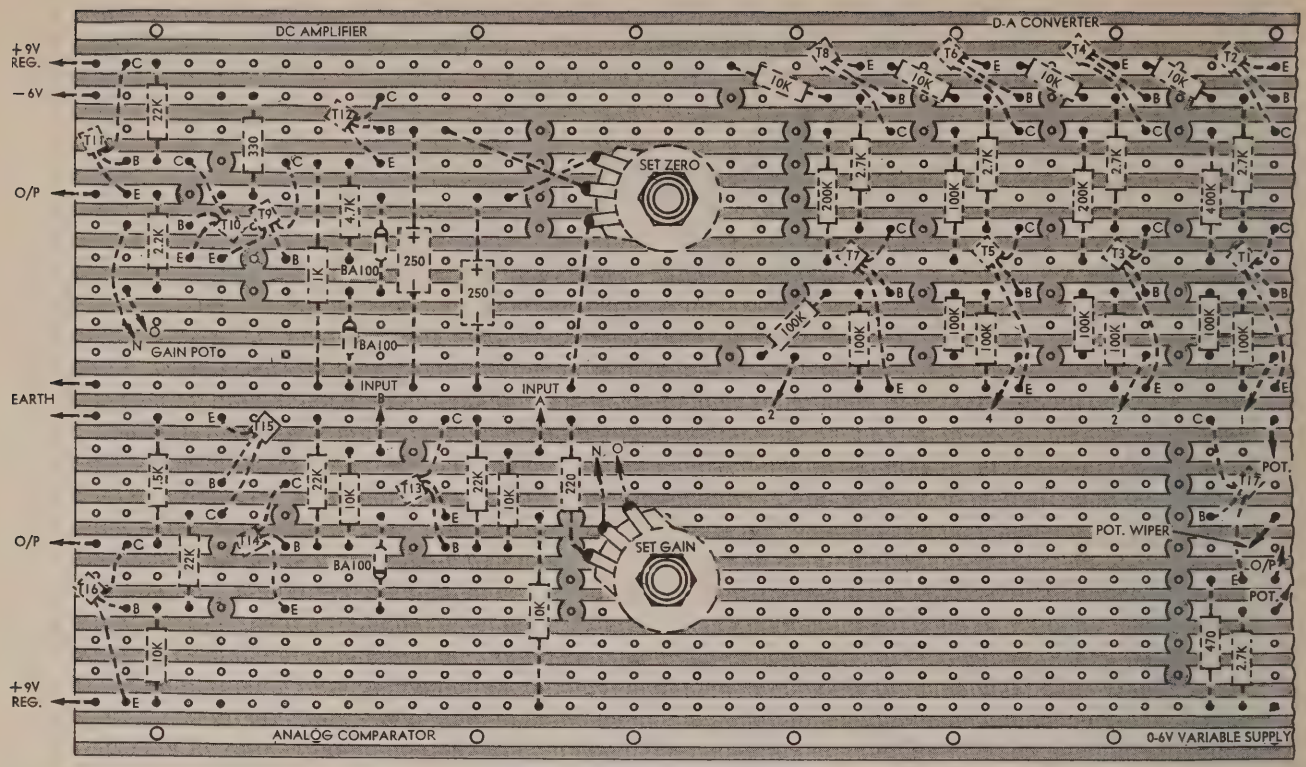
the drivers when the readout is disabled.

The readout unit employed is of the edge-lit engraved plastic sheet variety, employing miniature 12V 100mA lamps. Designated the Koten RK-E1, it may be ordered via the usual parts suppliers from Craftsman Glass Pty. Ltd., of 460 Bexley Road, Bexley, N.S.W. This unit is particularly suited to the demonstrator because of its modest cost, small physical size and low-voltage operation.

As may be seen from the photograph of the rear of the third panel, the decoding matrix and readout drivers are wired using one half of a full-length 4/1001 Vero-board strip. The wiring of the prototype is shown in figure 7 for those who wish to duplicate the arrangement used. Note that the diodes in series with the readout lamps are not mounted on the strip, but form the connections between the strip and the readout unit.

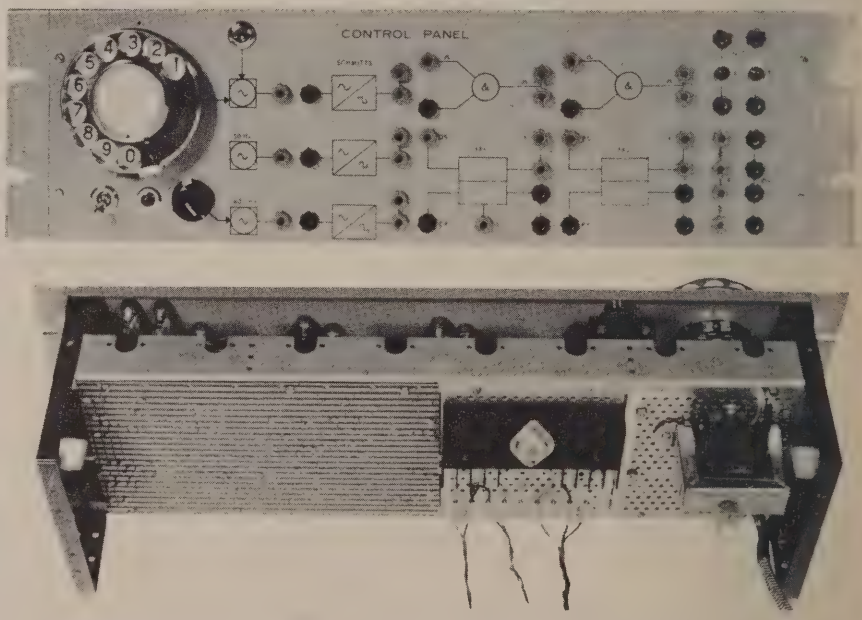
The conversion section of the third panel comprises a digital-to-analog converter (DAC) for 2421 BCD, an analog comparator and a small 0-6V variable DC supply used to provide an analog test signal. The circuitry involved is shown in figure 8. As may be seen the emphasis is again on circuit simplicity and the use of low-cost components.

The DAC circuit is of the simple SPST weighted-resistor type, using tran-



WIRING OF CONVERSION CIRCUITRY

Above is the wiring pattern used for the conversion circuitry, which occupies the right half of the third panel wiring strip.



Front and rear views of the fourth and final panel, which provides circuitry for timing, control and power supply.

sistor switching from positive input logic. Transistor pairs T1-T2, T3-T4, T5-T6 and T7-T8 are the input driver-switch combinations, and in each case the second transistor of the pair is turned "on" when a logical 1 (+9V) is applied to the input terminal.

The emitters of the switch transistors connect to the regulated +9V line, which acts as the reference voltage, and the collectors connect to the appropriate high-stability resistors R1, R2, R3, and R4. Hence each digital input is responsible for an appropriate increment in the total current nI passed by the four resistors.

Transistors T9-T12 form a simple DC amplifier which derives from the basic DAC output current a proportional voltage ranging from 0-5.5V DC. The output voltage is monitored by a small meter and is made available at a panel jack for inspection on a CRO, connection to the analog comparator, and so on.

Transistor T9 is an input emitter follower, in "inverted" configuration to give a low "offset." The latter term describes a range in input quantity values which is treated by a device as if all the values included do not differ from zero. If a standard emitter follower were used offset would tend to be high due to the turn-on voltage of the base-emitter junction.

Transistor T10 is a feedback mixer and voltage amplifier used to stabilise and adjust the gain. T11 is an output emitter follower, while T12 is used to provide an adjustable bias on T9 for zero setting the analog output.

The analog comparator is again a simple circuit, using four low-cost transistors. T13 and T14 are input emitter followers, again of the "inverted" type to provide low offset. Silicon PNP transistors are used because of their low leakage.

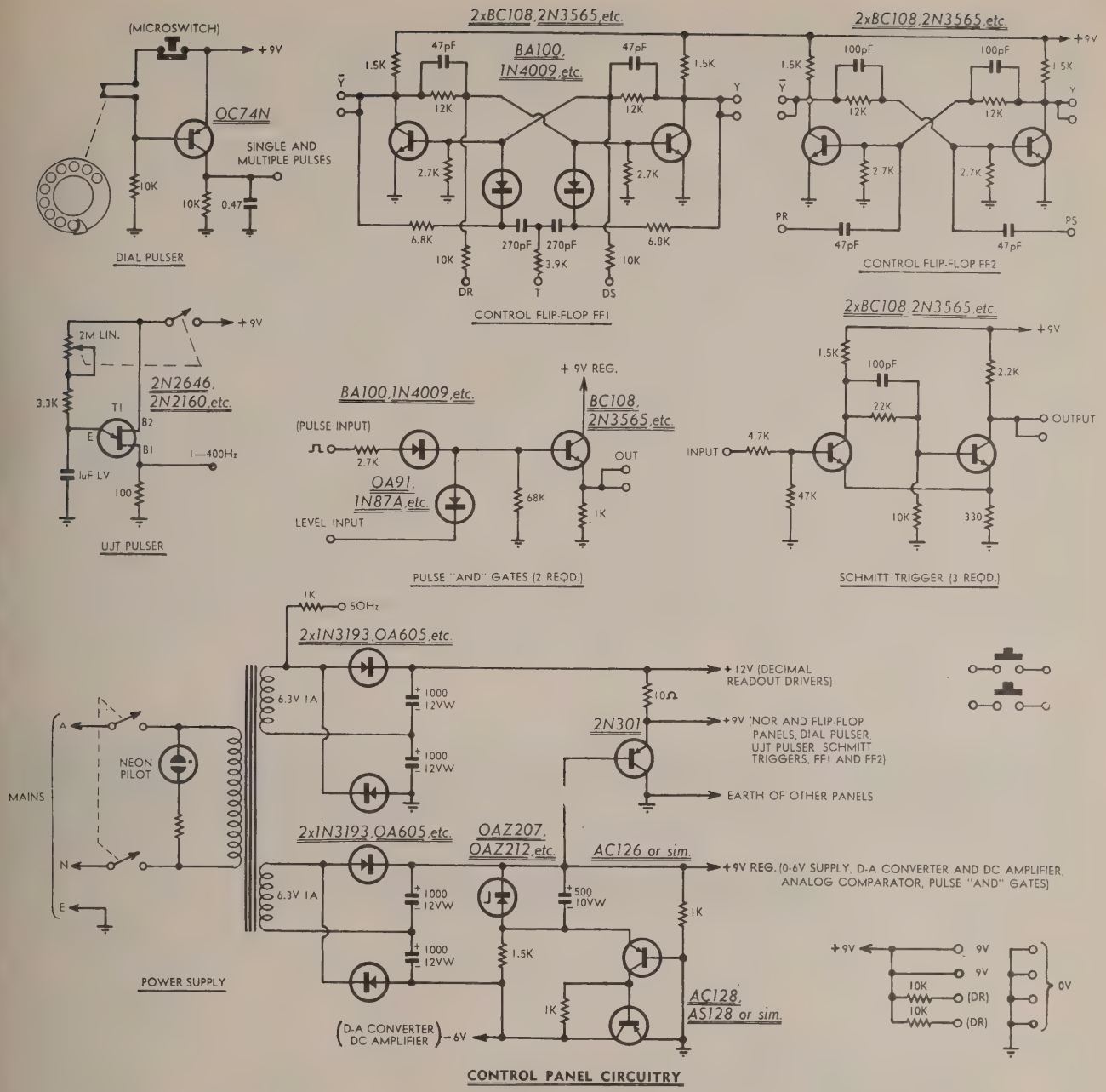
T13 and T14 share a common emitter resistor, T14 being coupled to the resistor via a diode D1 which serves to provide it with a small initial hold-off bias. The collector of T14 is coupled to the base of T15, which in turn couples to output transistor T16 to give a high voltage amplification.

Transistor T15 is a silicon planar NPN type BC108, 2N3565 or similar. T16 may be either a silicon planar type 2N3638A or a germanium type AC126 or similar. Diode D1 should be a silicon type BA100, 1N4009 or similar.

The 0-6V analog supply is a simple inverted emitter follower arrangement using a silicon 2N3638A or a germanium OC74N or similar.

As may be seen from the photograph the four BCD input jacks are in a horizontal row near the top centre of the panel, with the meter and DAC analog output jack at the right. Beneath the meter is the comparator symbol and its input and output jacks, while the control knob and jack for the 0-6V supply are to the left of the comparator. Black banana jacks are used for the DAC analog output and the negative comparator input, and red jacks for the remaining connections.

The wiring of the conversion circuitry is shown in figure 9 for those who wish to duplicate the prototype. Note that the two preset pots for



Above is the circuitry used in the fourth panel. It includes pulse generators, Schmitt triggers, gates and control flip-flops.

the DAC amplifier are mounted on the wiring strip between the analog comparator and 0-6V supply wiring.

The fourth and lowest panel of the demonstrator is that concerned with timing, control and power supply. It provides fixed 50Hz and variable 1-400Hz pulse generators, a telephone dial/pushbutton pulser, three Schmitt triggers, two pulse AND gates, two control flip-flops, two "floating" push-buttons, and a power unit which delivers supply voltages for the complete demonstrator.

Figure 10 shows the circuitry involved in the fourth panel, while the layout of the panel may be seen in the photograph.

A unijunction relaxation oscillator is used to generate pulses which may be varied from 1-400Hz. The unijunction employed may be either of the economy types 2N2646 or 2N2160.

An OC74N or similar PNP germanium medium-power type is used to generate pulses from the telephone dial or the microswitch pushbutton. The 0.47uF capacitor in the collector circuit of the transistor is for suppression of spurious pulses due to contact bounce.

Telephone dial mechanisms may be obtained at modest cost from disposals sources. The contacts to use in this circuit are those which are normally closed, opening only for the actual pulsing. Note that the push-button used for single pulsing should be of the microswitch variety for reliable operation.

Fixed 50Hz pulses are made available at a panel terminal via a resistor connected to one secondary of the power transformer.

Three Schmitt triggers are provided for squaring-up signals from the pulse sources and from external sources of signals. All three are identical and wired to the simple circuit shown. Note that the input circuit contains no protection, so that if external sources of signal are used with the demonstrator they should be limited in amplitude to less than $\pm$ 4 volts.

Two pulse AND gates are provided to serve as signal gates for counter, frequency meter, ADC and similar configurations. Both are identical and wired to the simple circuit shown. The logic polarity at the control input is positive: the gate is "open" when the control input is at +9V, and "closed" when it is at 0V.

Two control flip-flops are provided both for signal-gate control and as demonstation units in their own right. One (FF1) is a gated R-S type wired internally for toggling-mode operation, while the other (FF2) is a simple R-S type fitted only with pulse set (PS) and reset (PR) inputs. Using these it is possible to set up almost all of the standard control configurations.

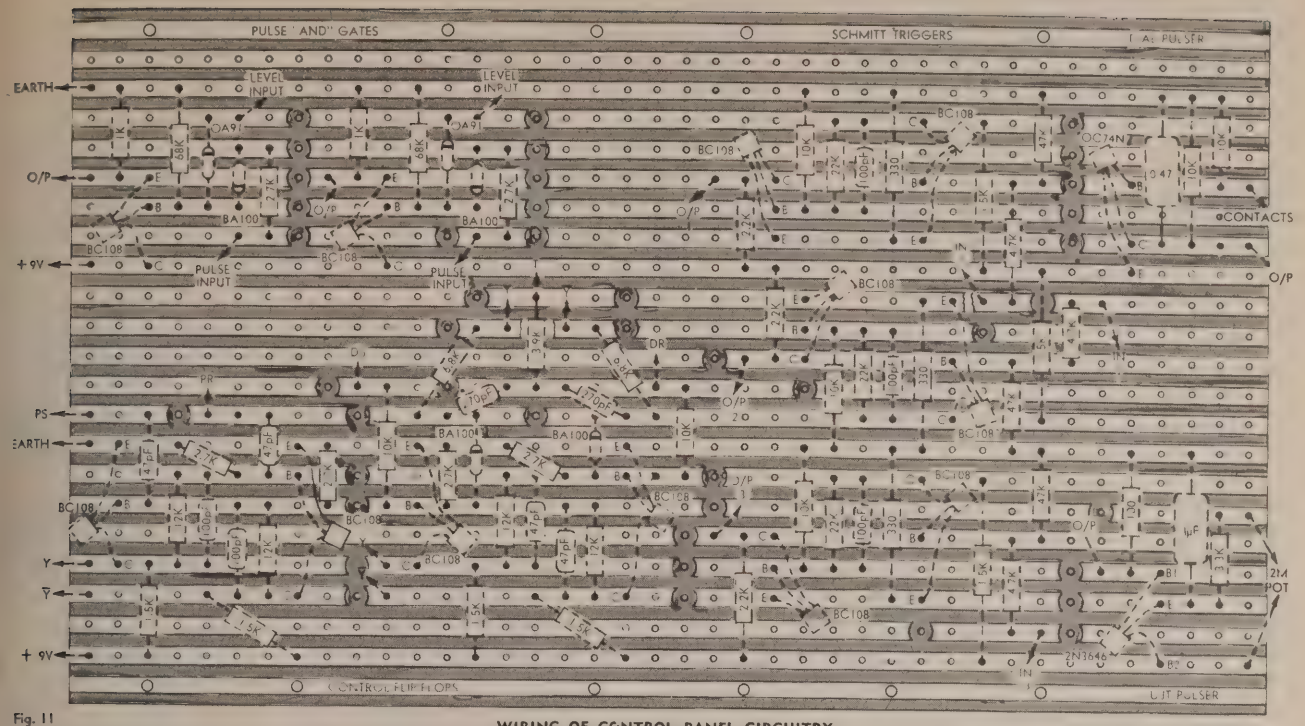


Fig. 11

WIRING OF CONTROL PANEL CIRCUITRY

The wiring pattern used on the Vero-board strip of the fourth panel. Power supply wiring is simple and has not been shown.

The power supply unit consists of two separate voltage doubler circuits using low-voltage silicon diodes. A series regulator circuit connected to one rectifier is used to provide both +9V for the conversion circuitry and approximately −6V for the zero set circuit of the DAC output amplifier. The second rectifier circuit delivers +12V for the readout driver amplifiers, together with approximately +9V for the remainder of the demonstrator. The second +9V supply is provided by means of a shunt regulator which is referenced to the first supply.

As with the first three panels, the fourth panel employs simple circuitry and low-cost components. In all cases the aim has been to provide satisfactory performance at minimum expense. With the exception of the power supply circuitry the wiring of the fourth panel is achieved using an 8-inch length of 4/1001 Vero-board panel. A wiring diagram for the panel is shown above for the benefit of those wishing to duplicate the original. No wiring diagram is shown for the power supply as this wiring is sufficiently simple and non-critical for the constructor to devise conveniently his own layout on the two LK-111 chassis plates.

On the front of the fourth panel red banana jacks are used for the pulser outputs, the Schmitt outputs, the pulse inputs and the outputs of the pulse AND gates, the DS, T and Y connections for FF1, the PS and Y connections for FF2, and the four +9V supply outputs (two of which have 10K isolation resistors for direct resetting of flip-flops, etc.). Black jacks are used for the remaining connections. Note that a double-pole mains switch and a neon pilot lamp are mounted on the panel beneath the telephone dial. Note also that multiple output jacks are fitted to the Schmitt triggers, the pulse AND gates and the control flip-flops; this makes for convenient interconnection with the more complex configurations.

This completes the description of the demonstrator itself. The following and final chapter discusses some of the many digital operations and configurations which it may be used to demonstrate.

REFERENCE

GRAY, S. B., "A Survey of Digital-Logic Training Devices," in **Electronics**, V.37, No. 23 (August 24, 1964).

Chapter 13

Using the Demonstrator

In this chapter the author concludes the description of his demonstrator unit by giving a brief idea of the many digital principles and configurations which it may be used to demonstrate.

It should be noted right from the outset that the discussion which follows in this chapter will give only a brief idea of the tuitional potential of the demonstrator unit. There appears to be a very large number of configurations within the unit's "repertoire," and in the space available it will only be possible to give the reader but a sampling of these.

Although this is the case, intending constructors need have little fear that, having constructed the unit and tried the setups described, they will be at a loss to think of further configurations. The author can testify from personal experience that the insight gained from even a short "play" with the basic configurations is usually more than sufficient to suggest many others.

Before commencing the discussion itself it should scarcely be necessary to mention that the constructor or demonstrator will have to prepare quite a large number of patch-cords, with which to perform the various interconnections.

The cords should preferably be of various lengths and colours to ensure neatness and easy identification. They are constructed quite simply by fitting banana-type plugs to each end of flexible (multi-strand) plastic covered wire.

The author has also found it worthwhile to make up a few "cheater" cords with an alligator clip at one end, in order that for certain of the more complex configurations one is able to make multiple connections to a single panel jack.

Certain configurations will also require the use of "outboard" components, and these will be mentioned at the appropriate point in the discussion. It is suggested that the more common of these be wired to small tagstrips or resistor panels and fitted with connecting leads and banana plugs for convenience in setting-up.

The discussion itself will now follow, dealing first with the basic operation of the logic elements and then with appropriate configurations.

LOGIC GATES: Although the six gates provided on the top panel are labelled "NOR" elements, as we have seen before, interpretation of the logical function of an element or circuit depends wholly upon the logic polarity conventions adopted at input and output.

To show this, connect one side of each of the two push-buttons provided on the fourth panel to +9V, and connect the remaining jacks of the buttons to two inputs of one of the gates. Then by pressing either button the gate indicator lamp should extinguish.

If the inputs are interpreted as "true" when the buttons are in the depressed state and the gate output is interpreted as "true" when the lamp is lighted, the gate is fairly clearly performing the NOR operation. But consider what happens if the undepressed state of the buttons is regarded as corresponding to "true" inputs: now the gate is seen to perform the AND operation. And one can just as easily interpret the action of the gate as the operations OR or NAND, by simply reversing the output polarity convention so that the "true" state corresponds to the lamp being extinguished.

Similarly, if a single pushbutton and input are used, the gate may be regarded as either an inverter or NOT gate or a non-inverting driver and indicator.

This simple demonstration should serve very well in demonstrating the truth of deMorgan's theorem and the full significance of logic polarity convention. However, there may be occasions when it will be desirable to be able to "concoct" the various elementary operations so that "true" inputs are indicated by a depressed button and "true" outputs by a lighted lamp.

In general, it will be found that this presents no problems, as one can always synthesise any elementary operation using a single logical convention and a suitable number of NOR gates. Thus, for example, one can synthesise an AND gate using two gates simply as inverters between the buttons and the final NOR element; the three gates together then perform the AND operation in positive logic.

Apart from this, it will be found possible in many cases to perform wiring inversion. Hence by connecting two inputs of a gate to the fourth panel terminals supplying +9V via protective resistors, and wiring the buttons so that they shunt the supply terminals to ground, the buttons will both have to be depressed before the gate lamp will light — so that the setup will again perform the AND operation. (But remember to use the supply terminals incorporating protective resistors!)

There are many logical configurations which may be synthesised using the six gates provided. Two interesting examples are the "exclusive-OR" and "equivalence" functions which were described in the second article of this series.

It may be recalled that a frequent use for the gates is as binary readout devices for display of the state of flip-flops and other elements. Although as readout devices the gates involve a polarity inversion, this is rarely of consequence, as the complement of the signal to be displayed will usually be readily available.

FLIP-FLOPS: The synthesis of a simple R-S flip-flop may be demonstrated by connecting together two of the NOR gates so that the output of each goes to one input of the other. Then the flip-flop formed may be switched from one state to the other by applying +9V to a second input on each gate via the two bush-buttons. It will be found that the flip-flop will "remember" which button was pressed last — a demonstration that a bi-stable element has "memory" and can be used for information storage.

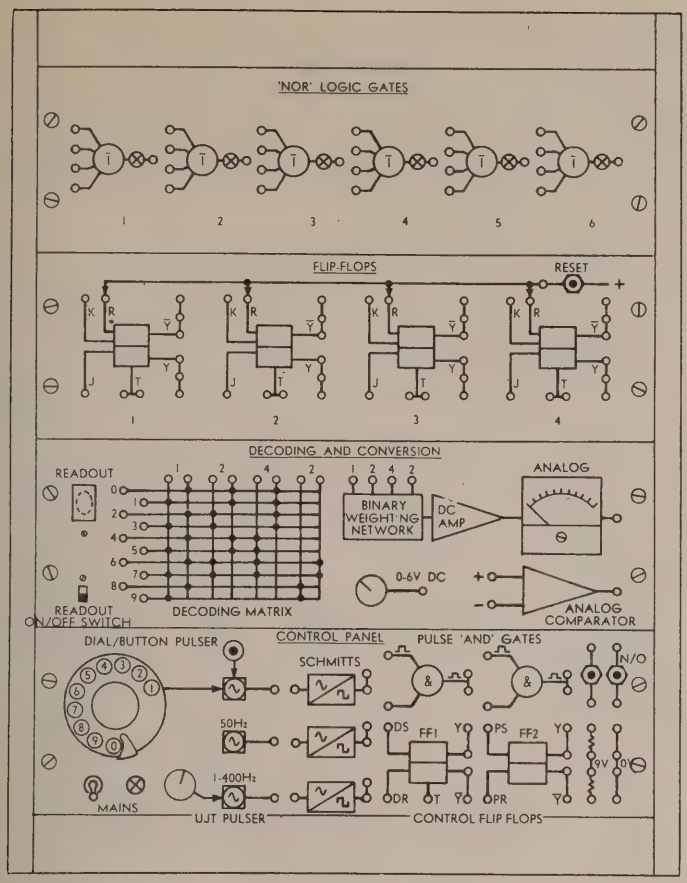
Flip-flop FF1 on the fourth panel may also be used to demonstrate simple R-S flip-flop action, by using a NOR gate to display its state and using the two push-buttons to alternately connect its DR and DS terminals to the +9V line. The NOR gate should be connected to the Y-component output if it is desired to make the lighting of the lamp correspond to the positive-logic set (Y=1) state.

Gated R-S flip-flop action may be demonstrated using control flip-flop FF2, in conjunction with the two pulse-AND gates. Connect the two pulse inputs (PR and PS) to the outputs of the gates, and connect the gating level terminals of the gates to the corresponding flip-flop output terminals — that of the gate serving the PS input to the Y output, and that of the gate serving the PR input to the Y-complement output. Then connect the pulse inputs of the gates to the output of one of the Schmitt triggers, and connect the Schmitt input to the dial/button pulser.

Pressing the microswitch button should then cause FF2 to alternately set and reset, showing the toggling mode of operation made possible by the gated R-S configuration.

Flip-flop FF1 is, in fact, a gated R-S flip-flop connected for toggling mode operation, and thus may also be used to demonstrate the foregoing action. This may be done simply by displaying its state using a NOR gate, while feeding pulses to its T terminal using the same dial/button pulser and Schmitt trigger combination.

In a similar fashion one of the flip-flops on the second panel may also be shown capable of toggling. If the flip-flops have been constructed as J-K types, this is done in identical fashion to that just described for FF1; if they have been constructed as gated R-S types it will be necessary to perform the small additional operation of interconnecting



The above diagram should serve as a guide in reading the text of this article, particularly for those who have not as yet constructed the unit.

the SG and Y terminals and the RG and Y-complement terminals.

COUNTING AND SCALING: Using either FF2, FF1 or a panel flip-flop in toggling mode it is also an easy matter to show that such an element can be used for x2 scaling and frequency division. Do this simply by connecting a second NOR gate to the Schmitt trigger output, and reconnecting the Schmitt input to the UJT pulser. The UJT pulser output frequency can then be set quite easily to show that the lamp connected to the Schmitt output is flashing at twice the frequency of that connected to the flip-flop output.

Two the second panel flip-flops may be connected in cascade to demonstrate a 2-bit or 0-3 binary counter which will also function as a x4 scaler and frequency divider. Do this simply by connecting both in toggling mode, connecting NOR gates as readouts, and connecting the T input of the second flip-flop to the Y output of the first. The T input of the first flip-flop becomes the "input," and may be fed pulses from either the UJT or dial/button pulsers via a Schmitt trigger.

By connecting the T input of the Y output of the first the pair will perform "up" counting—i.e., the states will be 00, 10, 01, 11, 00. To demonstrate "down" counting, simply change the carry-over connection so that it connects the second T input to the first Y-complement output. The counting sequence will then be 00,11,01,10,00.

Adding a third flip-flop and its readout will form a 3-bit or 0-7 binary counter, which may also be regarded as a single digit **octal** counter or as a x8 scaler/frequency divider. Again the carry-over connections may be altered to produce either "up" or "down" counting.

Adding the fourth flip-flop will produce a configuration which may be regarded as either a 4-bit or 0-15 binary counter or as a x16 scaler/frequency divider.

The last-named configuration may be arranged to perform **decimal** counting in 2421 BCD code by applying feedback between the Y-complement output of the fourth element and the DR terminals of the second and third elements so that the latter are re-set on the count of "8."

The feedback may be applied by connecting 100pF capacitors between the DR terminals and the fourth Y-complement terminal. It is suggested that the two capacitors be mounted on a small section of miniature resistor panel for protection, and that they be provided with connection leads and plugs for convenient connection into circuit.

DECODING AND READOUT: Decimal readout may be connected when the flip-flops are connected as a BCD counter, and this is done simply by connecting the eight decoder input terminals to the appropriate flip-flop output terminals and switching on the readout unit. However, before doing this it may be worthwhile to demonstrate the actual principles of decoding by using the NOR gates to individually decode one or two of the decimal digits. Reference to the seventh article in this series will provide guidance in this regard.

In passing it may be noted that the decimal decoding and readout circuitry may be used as an **octal** decoder/readout, simply by using only the first three pairs of decoder inputs and connecting the "Y" jack of the fourth pair to ground. Thus the decoder may be used to read-out the digits 0-7, corresponding to the 8 states of the 3-bit binary counting configuration.

All six flip-flops of the demonstrator may be connected in cascade to form a 6-bit or 0-63 binary counter, alternatively regarded as a 2-digit octal counter (00-77 octal counting) or a x64 scaler/frequency divider. This is done by connecting FF2 as a toggling element using the two pulse-AND gates, feeding its carry-over to FF1, and feeding the carry-over of FF1 to the cascaded flip-flops on the second panel.

The six NOR gates may be used as readouts, and/or the decoding and readout circuitry may be used to give an octal readout of three of the elements. Again "up" and "down" counting may be performed as desired simply by changing the carry-over links.

Note with regard to the above configuration that FF2 should be placed first in the chain. This is necessary because loading effects prevent the (FF2 + pulse-AND gates) combination from triggering reliably when fed from a flip-flop output; the combination must be fed from a Schmitt trigger output.

OVERFLOW AND OVER-RANGE: With the four flip-flops of the second panel connected for BCD counting, control flip-flop FF1 may be used to demonstrate the "over-range" facility provided on many digital instruments. This may be done simply by fitting FF1 with a NOR gate readout and connecting its T input to the Y output of the fourth element of the counting decade. It will be seen that the addition of the fifth flip-flop effectively doubles the capacity of the counter, which will now overflow on the pulse following the registration (9+10=19) rather than on that following (9).

In a similar fashion one can easily demonstrate the detection of counter "overflow" and the mechanism of input lock-out when overflow occurs. To do this simply connect the DS terminal of FF2 to the Y output terminal of the final counting flip-flop, and connect the Y-complement output of FF2 to one of the pulse-AND gates so that it controls the pulses fed to the counter input. Then if a NOR gate is connected also to the Y-complement output of FF2, it will be seen that as soon as the counter overflows, FF2 will set and the pulse-AND gate will be closed to prevent further pulses reaching the counter.

In order that the demonstration may be repeated it will be necessary to connect the PR terminal of FF2 to the dial/button pulser via one of the Schmitt triggers. The cycle of events will then be repeated each time FF2 is reset using the microswitch button.

Both over-range and overflow detection may be demonstrated by combining the two configurations just described. Use FF1 as the over-range flip-flop as before, and take the DS signal to FF2 from the former's Y output. Overflow will now occur on the arrival of the first pulse following the (9+10) registration and, as before, this will result in the setting of FF2 and consequent lock-off of the input.

SHIFT REGISTERS AND SHIFT COUNTING: The four flip-flops of the second panel may easily be used to

demonstrate the operation of the shift register and the normal and twisted ring counters. Basically, this involves interconnecting the Y and Y-complement outputs of each element to the K (or RG) and J (or SG) inputs, respectively, of the succeeding element, and application of the input pulses to all T inputs in parallel. The fifth article in the series should be consulted if guidance is required concerning the principle of operation.

To demonstrate the action of a shift register, do not make the final pair of interconnections between the output of the fourth element and the gating inputs of the first element. Instead, take the J and K (or SG and RG) terminals of the first element to a suitable combination of NOR gates, arranged so that by pressing a button or releasing it, either J or K may be taken to earth potential while its fellow is taken to +9V. (It will be convenient if the K or RG terminal is taken to +9V when the button is in the depressed state.)

If pulses are now fed to the flip-flops, either from the dial/button pulser or from the UJT pulser set for a slow rate, it will be seen that the elements will "shift" along the line the conditions at the first pair of input gating terminals, ultimately "pushing them out the end" at the output of the fourth element. Each input pulse will cause the input conditions to be shifted one element along the register.

To convert the shift register setup into a **normal ring counter** the main change required is to remove the "external" input to the first element and form the register into a closed loop by fitting the remaining connections between the output of the fourth element and the input of the first. However, as this only gives the register the ability to "pass a pattern around the ring," one must also ensure that there is in fact a pattern to be passed!

This may be done in two ways. The first and probably neater way is to logically "invert" the first element, so that the second panel reset button effectively "sets" this element while resetting the remaining three. This may be done simply by swapping all the connections between the Y and Y-complement output jacks, and between the J and K (or SG and RG) gating terminals.

The second way is to leave the connections to the first element as before, and to obtain a situation where one of more of the elements are set by temporarily wiring the register as a ripple-carry counter or a shift register. The connections are then changed to the ring counter configuration, and if required all but one of the elements may be reset by connecting their DR terminals to one of the **series-resistor** +9V supply jacks.

Either of the foregoing methods may be employed, but the first will tend to be more straightforward and less time-consuming. It will also serve to demonstrate that the labels given to flip-flop terminals are to a large extent arbitrarily assigned, depending upon the operation to be performed.

Demonstration of the **twisted-ring counter** is somewhat simpler than that of the normal ring counter, because the former generates its own pattern. All that is necessary is to "uncross" the final interconnection between the outputs of the fourth element and the gating inputs of the first (or to "cross" them if the first element has been inverted). Upon resetting the group and feeding pulses to the common T-line, they will then pass through the characteristic sequence of eight state combinations.

Decoding and readout of the three basic shift mode configurations is not easily possible using the circuitry on the third panel. However, as before, the NOR gates may be used either as simple binary readouts or to demonstrate the decoding of individual digits. In most cases simple binary readout should be quite adequate.

SIMPLE FREQUENCY METER: Using the four main flip-flops wired as a single decade BCD decimal counter, it is fairly easy to demonstrate the basic operation of a digital frequency/time period meter.

Do this by using one of the pulse-AND gates as a main signal gate to the counter, controlled by FF1. Then use the second pulse-AND gate as a timebase gate—connecting to the T terminal of FF1, and controlled in turn by FF2. Connect the PR terminal of FF2 to the Y output of FF1, so that when FF1 closes the main gate, FF2 is reset to close also the timebase gate. Then connect the dial/button pulser to a Schmitt trigger, and connect the trigger output to both the PS terminal of FF2 and the common reset terminal on the second panel.

If the UJT pulser and the 50Hz pulser (both fed through Schmitt triggers) are used as "input" and "timebase" signals, resetting the system via the microswitch button will cause it to indicate the frequency radio between the two. Hence, if the 50HZ signal is used as the timebase, the register will indicate the frequency of the UJT pulses in multiples of 50Hz; while with the opposite arrangement the indication will be equivalent to the period of the UJT pulses in multiples of 20mS.

In both cases it will be observed that there is a tendency for repeated readings to vary over three adjacent digits. This is due both to the inevitable one-pulse gating ambiguity, and to the lack of synchronisation on the timebase gating. In any case the measurement resolution will be very poor, because the basic counter consists of a single decade.

TIMER/DFM WITH SYNCHRONISED GATING: It is possible to demonstrate the more practical timer/frequency meter configuration employing synchronisation of the timebase gate, by reducing the actual counting register to a 3-bit binary or octal register (using the decoder for octal readout). This releases the fourth flip-flop and permits the latter to be used with FF1 and FF2 in the gating circuitry.

The new setup will be similar to that for demonstration of the simple configuration, except that the additional flip-flop will now control the timebase gate in place of FF2. In turn, FF2 will control the toggling of the new flip-flop via interconnections to the J and K (or SG and RG) terminals of the latter: connect Y to K, and Y-complement to J. Also connect the Y-complement output of FF2 to the DR terminal of FF1, and the T terminal of the additional flip-flop to the output of the Schmitt trigger driving the timebase gate. NOR gates may be connected to the Y-complement outputs of all three control elements to allow display of their states.

Upon reset of the setup using the microswitch button as before, it will be seen that the sequence of events followed is:—

(a) The register resets to 000 binary or 0 octal, and upon release of the microswitch FF2 sets.

(b) The additional flip-flop sets on the first timebase pulse to reach it after FF2 has set, and this opens the timebase gate.

(c) The next timebase pulse passes through the timebase gate and sets FF1, opening the main gate.

(d) The third timebase pulse again passes through the FF1, resets it, and thus indirectly resets FF2 as well. This places plus 9V on the J terminal of the additional flip-flop, and also on the DR terminal of FF1, so that—

(e) The fourth timebase pulse simply resets the additional flip-flop, closing the timebase gate.

This sequence of events is best seen by using the UJT pulser as the source of timebase signal and adjusting it for a very low output rate. The sequence will be repeated each time the microswitch button is depressed and released.

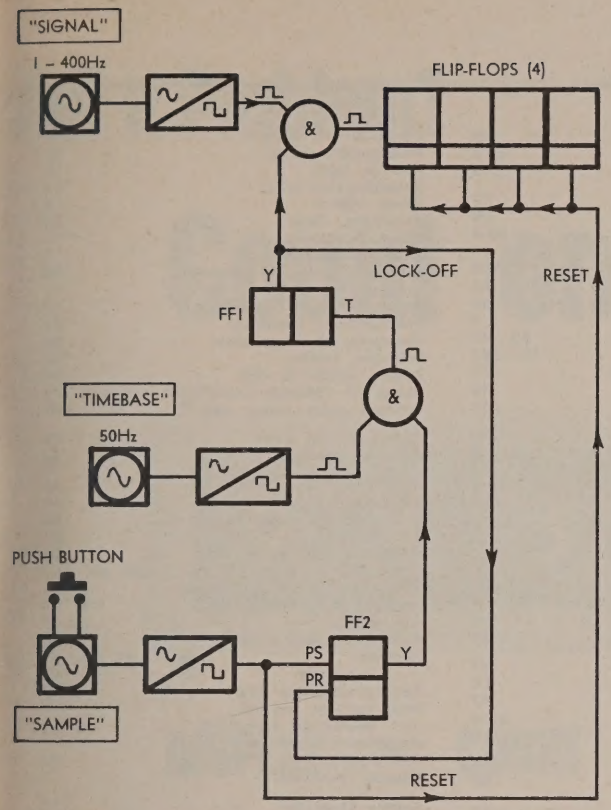
STORAGE-TYPE READOUT DISPLAY: The storage-type display used in the more elaborate digital instruments may be demonstrated by using the six flip-flops of the unit as two 3-bit binary or single-digit octal registers, one of which is used to store the count of the other using the technique known as **"jam transfer."**

To do this, connect FF2, FF1 and the first of the main flip-flops (in that order) in the manner already described as a normal ripple-carry 3-bit "up" counter, fitting them with NOR gates for readout and using the UJT pulser as the source of input signals. Then wire the remaining three flip-flops so that in each case their K and J (or RG and SG) gating inputs connect to the Y and Y-complement outputs (respectively) of the three counting flip-flops. Fit the remaining NOR gates —and the decoder/readout, if desired —to the second three elements, and connect their three T inputs together and to the dial/button pulser via a Schmitt trigger.

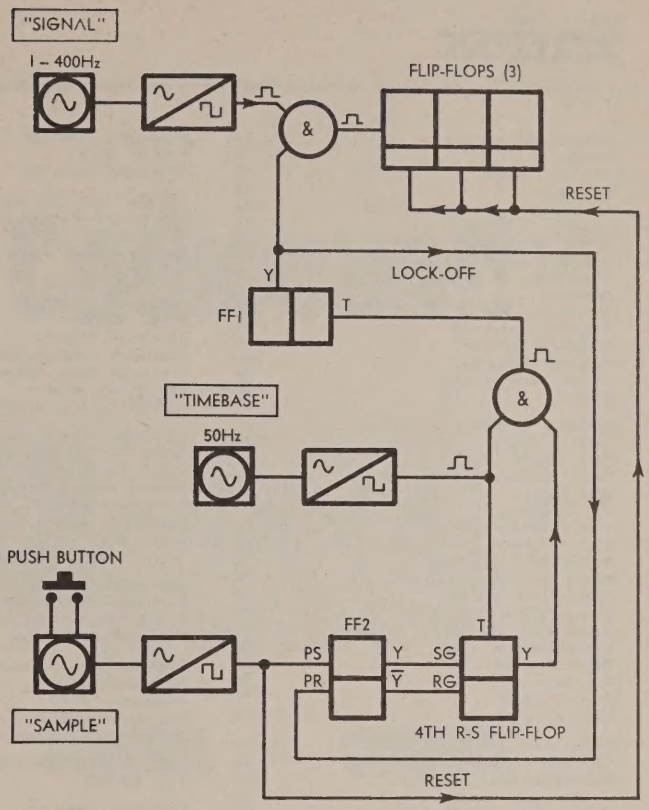
What the above connections do is produce what is effectively three two-element shift registers, the first element of each being the three counting flip-flops and the second element of each being the three storage flip-flops.

Hence at any stage of the counting being performed by the first three elements, a pulse applied to the T terminals of the second three (via the microswitch button) will force the latter elements to adopt the states currently present in the counting elements. This registration will then remain fixed in the second register until a further transfer is initiated by a second pulse applied to the common T line; the system thus demonstrates one common method of "storing" a counter registration upon command.

DIGITAL-TO-ANALOG CONVERSION: Demonstration of the operation of the DAC circuitry may be performed simply by connecting a lead between the plus 9V line and the 1, 2, 4 and 2 inputs in turn (the gain and zero set of the DC amplifier should be adjusted before this is done). It will be seen that the meter output will be directly



SIMPLE DFM



DFM WITH SYNCHRONISED GATING

The diagrams above should help in connecting up the demonstrator unit for the elementary "simple" and "synchronised gating" digital frequency meter configurations described in the text opposite. At left is the simple DFM configuration, while at right is the slightly more complex synchronised gating configuration.

proportional to the digital input terminal concerned, showing the weighting principle.

To show the DAC in actual operation it is simply necessary to wire up the four main flip-flops for BCD decimal counting, and then connect the DAC inputs to the appropriate Y outputs. If the count is proceeding at a suitably slow rate, the meter will then be seen to "step" in parallel with the numerical readout.

If the counting rate is increased, the meter will gradually adopt a fairly steady average reading. However, by connecting an oscilloscope to the analog output terminal it will be seen that the DAC is still producing a faithful "stair-step" output signal.

ANALOG-TO-DIGITAL CONVERSION

Operation of the analog comparator is easily demonstrated by connecting a NOR gate to the output of the element and the 0-6V supply—set to a convenient voltage — to the "+" input. Then by connecting the "—" input to earth and +9V in turn, it may be that the comparator output depends upon the relative magnitude of the two inputs.

A more dramatic demonstration of the comparator in action may be obtained by connecting the "—" input to the analog output of the DAC. Then by first resetting the counter and then feeding in single pulses from the dial/button pulser, the DAC output may be stepped until it just exceeds the voltage supplied by the 0-6V supply—whereupon the comparator output will change state.

By varying the output from the 0-6V supply and repeating the demonstration it will be seen that the comparator is capable of indicating on a "yes/no" basis the relative magnitudes of the two analog signals.

To demonstrate a complete ADC of the simple counter type it is then simply a matter of connecting the output of the comparator so that it controls one of the pulse-AND gates wired in series with the counter input. Thus the comparator is able to allow into the counter just sufficient input pulses to produce a registration whose analog equivalent is slightly greater than the "input" analog signal provided by the 0-6V supply.

It will be found that the comparator can cope with **increases** in the analog "input" on a continuous basis, letting additional pulses into the counter as required to balance the input and output. But if the analog "input" **decreases**, the comparator is unable to allow a corresponding change in the output. This would only be possible if one had a counter which would perform bi-directional counting.

To permit the setup to cope with a decrease in input it is therefore necessary to force it to take repeated samplings. This may be done by using the 50Hz pulser to provide the counter input pulses, and wiring the UJT pulser so that it resets the counter at a once-or-twice per second rate via the common reset line.

Each time the counter is reset, the comparator will reopen the gate to allow a fresh supply of pulses to enter. Thus the system will be able to cope with a reduction in analog input by arriving at a lower registration on the next sampling following the reduction.

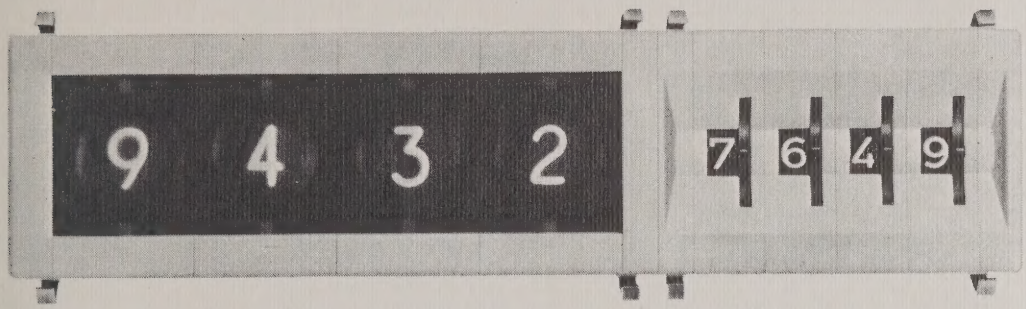
The configuration just decribed may be recognised as a simple type of potentiometric digital voltmeter. Naturally its accuracy will be very low with only a single counting decade—roughly plus or minus 20 p.c.—but it will serve to demonstrate the principle involved.

Well, there you have it — a basic repertoire of the digital principles which may be demonstrated with the unit. It is the hope of the author that in trying these out and developing more of their own, readers and constructors will experience as much painless (if not pleasant) self-tuition as he himself has experienced in developing the unit and writing this book.

Index

Got a Problem?

Count on Codicount!



Actual Size

DISPLAY • COUNT • CONTROL • COMPUTE

Developed by CONTRAVES to solve recurring problems in providing counter displays, CODICOUNT embodies a system based on combinations of displays, counters and other logical switch elements. All the components required for a counter element are assembled in a plastic housing. By combining several components, decimal figures of several places can be counted, displayed and processed.

CODICOUNT finds applications in:

Pulse control
Position control
Time measurement
Linear measurement
Speed measurement

Digital voltmeters
Core winding counters
Traffic counting
Piece counting
Numerical control

Nuclear Techniques
Data Processing
Weighing Techniques,
Dosing Techniques
Etc.

Arbor

SOLE AGENTS FOR AUSTRALIA AND NEW ZEALAND:

ARBOR PTY. LTD.

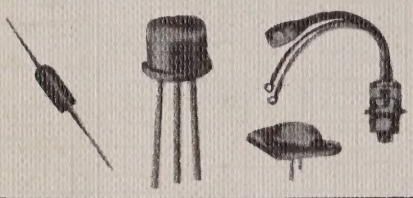
282 Bell Street, Coburg, Victoria, 3058
Phone 350-1144. Telegrams "Arbormel"

Printed and published by Sungravure Pty. Limited, of Jones Street, Broadway, at Morley Avenue, Rosebery.

Electronic Components for COUNTING

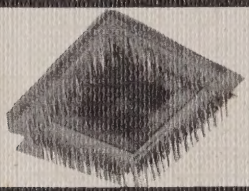
INTEGRATED CIRCUITS

Medium speed DTLs, for industrial and military applications. High speed TTLs and ultra fast E^2CLs for computer applications. Compatible, versatile linears for low level amplifiers, operational amplifiers for telecommunications, instrumentation and control.



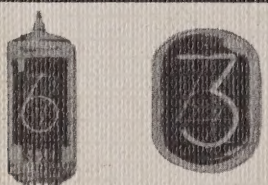
TRANSISTORS, DIODES, RECTIFIERS, THYRISTORS

Silicon planar, alloy, germanium alloy and alloy diffused transistors for general purpose, switching, low noise, low and high power applications. Silicon and germanium general purpose diodes, silicon zener diodes and power rectifiers. Thyristors for all industrial control applications.



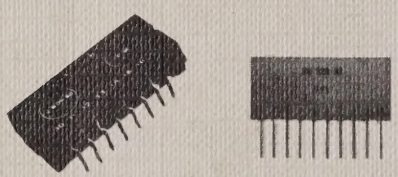
MEMORY PLANES, STACKS, CORES

Single plane matrices for low cost data storage. Lithium nickel ferrite cores permit operation over a wide temperature range. Available in various configurations for most applications. Complete range of miniature matrix stacks up to a capacity of 16,384 50—bit words.



DECADE COUNTER & NUMERICAL INDICATOR TUBES

A wide range of devices is available to suit most applications. Special quality characteristics ensure reliability and long life. Most indicator tubes can be obtained with a tinted filter for use under unfavourable ambient light conditions.



NORBIT 2, CIRCUIT BLOCKS

Mullard Norbit 2 includes sub-assemblies capable of solving industrial control problems using well established solid state techniques. Silicon semiconductors are used throughout so that an operating temperature range of from —10° C to 85° C is guaranteed, with speeds of up to 10 Kc/s. Each circuit block is a ready-made electronic sub-assembly designed to perform one of the basic functions in digital equipment. The range of circuit blocks includes pulse shapers, gate circuits, flip-flops, amplifiers, etc., to meet the requirements of digital systems engineers.

APPLICATIONS ENGINEERING SERVICE

This service operates in co-operation with, and at specific request of, commercial concerns requiring engineering assistance in the application of Mullard products. In addition, answers to technical enquiries are provided by the Technical Service Dept., where world-wide valve and semiconductor references are on file.

Mullard

Mullard-Australia Pty. Ltd.

35-43 Clarence St., Sydney, N.S.W. 2000. Phone: 29 2006 ● 123-129 Victoria Pde., Collingwood VIC. 3066. Phone: 41 6644 ● Cnr. Light St. and St. Paul's T'ce., Bowen Hills, QLD. 4006 Phone: 51 5151 ● 176 Wright St., Adelaide, 5000. Phone: 51 3123 ● 579 Murray St., Perth, W.A. 6000. Phone: 21 2561 ● Medhurst Wholesale Ltd., Hobart, Launceston, Burnie, TAS.

Associated with MULLARD LIMITED, LONDON